高等院校信息管理与信息系统专业系列教材

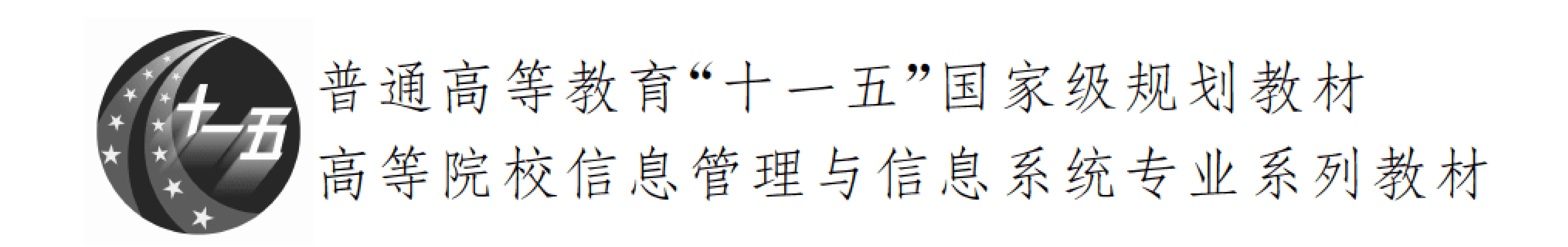
# 数据仓库与数据挖掘教程

## （第2版）

陈文伟 编著



清华大学出版社




普通高等教育"十一五"国家级规划教材
高等院校信息管理与信息系统专业系列教材

# 数据仓库与数据挖掘教程
# (第2版)

陈文伟　编著

清华大学出版社
北　京

## 内 容 简 介

数据仓库与数据挖掘是决策支持的两项重要技术，它们共同的特点是都需要利用大量的数据资源，并从数据资源中提取信息和知识。由于数据资源丰富，因此数据仓库与数据挖掘的决策支持效果显著。

本书系统介绍数据仓库原理，联机分析处理，数据仓库设计与开发，数据仓库的决策支持，数据挖掘原理，基于信息论的决策树方法，基于集合论的粗糙集方法、K-均值聚类、关联规则挖掘，仿生物技术的神经网络，遗传算法，公式发现，知识挖掘，文本挖掘与 Web 挖掘。

本书从数据仓库的兴起来说明决策支持的特点，从数据挖掘的理论基础来说明数据挖掘的方法，并通过实例来详细讲解。希望读者在学习之后，亲自在计算机上去实践，这样才能更有效地掌握数据挖掘的方法。



**图书在版编目（CIP）数据**

数据仓库与数据挖掘教程/陈文伟编著．—2 版．—北京：清华大学出版社，2011.11
（高等院校信息管理与信息系统专业系列教材）
ISBN 978-7-302-25913-8

Ⅰ．①数…　Ⅱ．①陈…　Ⅲ．①数据库系统－高等学校－教材 ②数据采集－高等学校－教材　Ⅳ．①TP311.13 ②TP274

中国版本图书馆 CIP 数据核字(2011)第 115742 号

**责任编辑**：白立军　李玮琪
**责任校对**：李建庄
**责任印制**：

**出版发行**：清华大学出版社　　**地　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　编**：100084
**社　总　机**：010-62770175　　**邮　购**：010-62786544
**投稿与读者服务**：010-62795954，jsjjc@tup.tsinghua.edu.cn
**质　量　反　馈**：010-62772015，zhiliang@tup.tsinghua.edu.cn
**印 刷 者**：
**装 订 者**：
**经　销**：全国新华书店
**开　本**：185×260　**印　张**：19.75　**字　数**：496 千字
**版　次**：2011 年 11 月第 2 版　**印　次**：2011 年 11 月第 1 次印刷
**印　数**：1～0000
**定　价**：　元

产品编号：040341-01

# 第2版前言

数据仓库(Data Warehouse,DW)和数据挖掘(Data Mining,DM)是决策支持的两项重要技术。在数据仓库中利用多维数据分析来发现问题,并找出产生的原因,能从大量历史数据中预测未来;利用数据挖掘方法能从大量数据中获取知识。两项技术的共同特点是都需要利用大量的数据资源。

数据仓库和数据挖掘是在20世纪90年代中期兴起的,经过十多年的发展,在技术和应用两个方面都得到了很大的提高。为了提高数据仓库的决策支持效果,近年来开展了对综合数据的数据立方体的压缩技术研究,以及对多维数据分析的MDX语言的推广。本书第2版增加了这两项内容。为了强化数据挖掘中神经网络与遗传算法两项实用技术,在第2版中把它们独立列为两章。在神经网络中,按从易到难的顺序将内容重新安排了一下,并增加了径向基函数网络RBF的内容。在遗传算法中增加了进化计算的内容,以便扩大读者的视野。

本书仍保留了按数据仓库的形成过程来讲述其内容的方式,即从数据库到数据仓库以及对比,从联机事务处理OLTP到联机分析处理OLAP以及对比,用它们的对比来突出数据仓库决策支持的作用。按形成过程来讲述,既有利于掌握它们的连贯性,又有利于掌握数据仓库的新特点。

本书保留了依照数据挖掘的理论基础来讲述数据挖掘的方法:大家熟悉的决策树方法实质上是利用信息论中计算信息量的公式来选择属性构造决策树的结点;影响较大的粗糙集方法是典型的利用集合的覆盖原理;关联规则挖掘方法是对相关事务(项)的子集占整个集合的比例,大于阈值时建立关联规则的;在集合论方法中增加了影响最大的K-均值聚类方法。读者在懂得数据挖掘的方法的理论基础后,能够更好地掌握和使用这些方法。

本书第12章由原来的第12章的“数据仓库与数据挖掘的发展”变为“知识挖掘”,这一章是全新的内容。第13章做了部分修改,增加了“Web日志分析与实例”一节。

作者从事数据仓库与数据挖掘研究工作多年,在本书第12章中介绍了作者完成的项目——“软件进化规律的知识挖掘”,相信能对本科生有启发作用。掌握这些软件进化规律,一来能够帮助学员提高软件使用能力;二来能够引起他们的兴趣,再进一步去挖掘软件进化规律,促进软件进化。本书中也介绍了作者领导的团队完成的项目:IBLE决策规则树方法、FDD公式发现系统、遗传分类学习系统GCLS、变换规则的知识挖掘等。这些内容并不要求本科生掌握,关键在于启发他们如何去创新。这些内容更适合研究生学习和相关行业的工作人员参考。

建议在本科教学中,对信息论原理、集合论方法、神经网络和遗传算法,只讲公式和应用,概略地说明原理的深层内容和公式的推导。这些知识的详细内容适合于研究生教学。

王珊教授曾说过:我觉得数据仓库或者数据挖掘,有时候挖掘出来的东西并不是很有用的,可能要经过很长时间,也许在某些情况下得到一个非常好的结果,能够给领导者一个启

示。但是不会像宣传的那样，我们今天建立了数据仓库系统，明天就能够解决商业竞争中的很多问题，就能取得很大的效益。而且，领导者的素质也是一个重要因素。领导者能不能发现这些问题，技术人员给他的新提示他能不能接受，数据挖掘对他是否有效，等等。这些问题都影响了数据仓库和数据挖掘的效果。

这段话说明了一个问题，数据仓库和数据挖掘的应用比技术有时显得更重要。作者也希望学员在学习这门课程时，除学习原理与技术外，还要加强应用能力的锻炼，即通过计算机去亲自实现它，体会它的真正价值。

欢迎广大读者与作者进行交流，为促进我国数据仓库和数据挖掘的发展而共同努力。

陈文伟

2011 年 9 月于广州

# 第1版前言

数据仓库(data warehouse,DW)是利用数据资源提供决策支持。它比利用模型资源辅助决策更有效,而且辅助决策的范围更宽。由于在现实中,数据大量存在,而且在迅速地增长,只要将面向应用(事务驱动)的数据库重新组织转变为面向决策分析的数据仓库,就可以帮助决策者从不同的视角,通过综合数据分析掌握现状;通过多维数据分析发现各种存在的问题;通过对数据层次的钻取找出问题产生的原因;通过历史数据预测未来。由于数据仓库辅助决策效果明显,数据仓库已经从20世纪90年代中期兴起,经过几年的发展,迅速形成了潮流。

数据挖掘(data mining,DM)是从数据中挖掘出信息和知识,是从人工智能的机器学习(machine learning,ML)中发展起来的。机器学习是让计算机模拟人的学习方法获取知识。机器学习中的大量学习方法已经引入到数据挖掘中。数据挖掘也是20世纪90年代中期兴起的。正是由于数据挖掘具有获取知识的能力,目前各数据仓库均将数据挖掘作为数据仓库的前端分析工具,用于提高数据仓库的决策支持能力。

数据仓库、数据挖掘和联机分析处理(on line analytical processing,OLAP)结合起来的新决策支持系统是以数据驱动的决策支持系统。而传统决策支持系统(decision support system,DSS)是以模型和知识驱动的决策支持系统,是由模型库系统、知识库系统、数据库系统和人机交互系统组成的。新决策支持系统利用的是数据资源,而传统决策支持系统利用的是模型资源和知识资源,它们两者辅助决策的方式和效果均不相同。新决策支持系统并不能代替传统决策支持系统,它们是相互补充的。新决策支持系统与传统决策支持系统结合起来形成的综合决策支持系统将是决策支持系统发展的新方向。

数据仓库、数据挖掘、联机分析处理等结合起来也称为商业智能(business intelligence,BI)。商业智能是一种新的智能技术,区别于人工智能(artificial intelligence,AI)和计算智能(computational intelligence,CI)。人工智能采用的技术是符号推理,符号推理过程形成了概念的推理链。计算智能采用的技术是计算推理,模拟人和生物的模糊推理、神经网络计算和遗传进化过程。商业智能是从数据仓库和数据挖掘中获取信息和知识,对变化的商业环境提供决策支持。商业智能是目前企业界正在大力推广的知识管理(knowledge management,KM)的基础。

作者于1997年6月30日在《计算机世界》报上发表了一组关于数据开采(数据挖掘)的文章,最早向国内学者介绍了数据挖掘概念和技术。作者又于1998年6月15日在《计算机世界》报上发表了一组关于数据仓库与决策支持系统的文章,在介绍基于数据仓库的决策支持系统上,提出了将基于数据仓库的决策支持系统和传统决策支持系统结合的综合决策支持系统,在国内产生了一定的影响。

本书的特点是从数据仓库和数据挖掘的兴起与演变来说明它们的本质,通过例子来解释它们的原理,既系统地介绍了数据仓库和数据挖掘的概念和技术,又介绍了它们之间的关

系，以及今后的发展。

在数据仓库的章节中，重点介绍数据仓库原理、联机分析处理、数据仓库设计与开发、数据仓库的决策支持应用。在数据挖掘的章节中重点介绍信息论方法、集合论方法、公式发现、神经网络和遗传算法，这些数据挖掘方法在现实中应用较广泛。由于数据挖掘的基础理论涉及面较宽，建议在本科生教学中对信息论原理和集合论方法只讲定义和例子，对神经网络和遗传算法只讲公式和应用，省略原理的深层内容和公式的推导。这些省略的内容适合研究生教学。

由于作者从事数据仓库与数据挖掘工作多年，并得到过国家自然科学基金项目的资助。在书中还介绍了作者领导的课题组完成的IBLE决策规则树方法、FDD公式发现系统、遗传分类学习系统GCLS等。本书也包含了作者提出的综合决策支持系统概念和可拓数据挖掘概念及理论，这些内容适合研究生学习和参考。

欢迎和广大读者进行交流，共同为促进我国数据仓库和数据挖掘的发展而努力。

参加本书录入的有毕季明、廖建文、赵健、徐怡峰、田昊等同志，在此表示感谢！

陈文伟

2006年5月29日于广州

# 目　录

# 第1章　数据仓库与数据挖掘概述

## 1.1　数据仓库的兴起

### 1.1.1　从数据库到数据仓库

由数据库发展到数据仓库，主要特征有如下几点。

- 数据太多，信息贫乏(Data Rich，Information Poor)。随着数据库技术的发展，企事业单位建立了大量的数据库，数据越来越多，而辅助决策信息却很贫乏，如何将大量的数据转化为辅助决策信息成为了研究热点。
- 异构环境数据的转换和共享。随之各类数据库产品的增加，异构环境的数据也逐渐增加，如何实现这些异构环境数据的转换和共享也成为了研究热点。
- 利用数据进行事务处理转变为利用数据支持决策。数据库用于事务处理，若要达到辅助决策的目的，则需要更多的数据。例如，利用历史数据的分析来进行预测，对大量数据的综合得到宏观信息等，都需要大量的数据。

数据仓库概念提出后，在短短几年的时间内就得到了迅速的发展。数据仓库产品也不断出现并陆续进入市场。

#### 1. 数据库用于事务处理

数据库存储大量的共享数据，作为数据资源用于管理业务中的事务处理。它已经成为了成熟的信息基础设施。

数据库中存放的数据基本上是保存当前的数据，随着业务的变化再随时更新数据库中的数据。例如，学生数据库，随着新生的入校，数据库中要增加新学员的数据记录。随着毕业学生的离校，数据库中要删除这些学员的数据记录。数据库总是保存当前的数据记录。

不同的管理业务需要建立不同的数据库。例如，银行中储蓄业务要建立储蓄数据库，记录所有储蓄用户的存款及使用信息。信用卡业务要建立信用卡数据库，记录所有用户信用卡的存款及使用信息。贷款业务要建立贷款数据库，记录所有贷款用户的贷款及使用信息。

数据库是为满足事务处理需求而设计和建立的，从而使计算机在事务处理上发挥了极大的效果。但是，数据库在帮助人们进行决策分析时就显得不适用了。例如，银行想了解用户的经济状态(收入与支出情况)以及信誉情况(是否超支，还贷情况等)，决定是否继续贷款给他，单靠一个数据库是无法完成这种决策分析的。必须将储蓄数据库、信用卡数据库、贷款数据库集中起来，对某一个人进行全面分析，才能准确了解他的存款及收支情况、信用卡使用情况以及贷款及还贷情况。这样，银行才能有效地决定是否给此人继续贷款。

同时使用三个数据库进行操作并非是一件简单的事，由于三个管理业务各自独立，在建立数据库时对同一个人可能使用了不同的编码，对于他的姓名可能有的用汉字，有的用汉语

拼音,有的用英文。这为使用三个数据库共同进行决策分析带来了困难。

### 2. 数据仓库用于决策分析

随着决策分析需求的扩大,兴起了支持决策的数据仓库。它是以决策主题需求集成多个数据库,重新组织数据结构,统一规范编码,使其有效地完成各种决策分析。

从数据库到数据仓库的演变,体现了以下几点:

(1) 数据库用于事务处理,数据仓库用于决策分析。

事务处理功能单一,数据库完成事务处理的增加、删除、修改、查询等操作。决策分析要求数据较多。数据仓库需要存储更多的数据,它不需要修改数据,它主要从大量数据中提取综合信息以及利用历史数据的规律得到预测信息。

(2) 数据库保持事务处理的当前状态,数据仓库既保存过去的数据又保存当前的数据。

数据库中的数据随业务的变化一直在更新,总保存当前的数据,如学生数据库、财务数据库等。数据仓库中的数据不随时间变化而变化,但它保留大量不同时间的数据,即保留历史数据和当前数据。

(3) 数据仓库的数据是大量数据库的集成。

数据仓库的数据不是数据库的简单集成,而是按决策主题,将大量数据库中的数据进行重新组织,统一编码进行集成。

如银行数据仓库数据是由储蓄数据库、信用卡数据库、贷款数据库等多个数据库按“用户”主题进行重新组织、编码和集成而建立的。

可见,数据仓库的数据量比数据库的数据量大得多。

(4) 对数据库的操作比较明确,操作数据量少。对数据仓库操作不明确,操作数据量大。

一般对数据库的操作都是事先知道的事务处理工作,每次操作(增加、删除、修改、查询)涉及的数据量也小,如一个或几个记录数据。

对数据仓库的操作都是根据当时决策需求临时决定而进行的。如比较两个地区某个商品销售的情况。该操作所涉及的数据量很大,不是几个记录数据,而是两个地区多个商店的某商品的所有销售记录。

### 3. 数据库与数据仓库的对比

数据库与数据仓库的对比如表 1.1 所示。

**表 1.1 数据库(DB)与数据仓库(DW)对比**

| 数据库(DB) | 数据仓库(DW) |
|---|---|
| 面向应用 | 面向主题 |
| 数据是详细的 | 数据是综合的和历史的 |
| 保持当前数据 | 保存过去和现在的数据 |
| 数据是可更新的 | 数据不更新 |
| 对数据操作是重复的 | 对数据的操作是启发式的 |

续表

| 数据库(DB) | 数据仓库(DW) |
| --- | --- |
| 操作需求是事先可知的 | 操作需求是临时决定的 |
| 一个操作存取一个记录 | 一个操作存取一个集合 |
| 数据非冗余 | 数据时常冗余 |
| 操作比较频繁 | 操作相对不频繁 |
| 查询基本是原始数据 | 查询基本是经过加工的数据 |
| 事务处理需要的是当前数据 | 决策分析需要过去和现在的数据 |
| 很少有复杂的计算 | 有很多复杂的计算 |
| 支持事务处理 | 支持决策分析 |

### 1.1.2 从 OLTP 到 OLAP

#### 1. 联机事物处理

联机事物处理(On Line Transaction Processing,OLTP)是在网络环境下面向交易的事务处理,利用计算机网络技术,以快速的事务响应和频繁的数据修改为特征,使用户利用数据库能够快速地处理具体的业务。其基本特征是用户的数据可以立即传送到计算中心进行处理,并在很短的时间内给出处理结果。这样做的最大优点是可以实时地处理用户的输入的数据,及时地回答。这样的系统也称为实时系统(Real time System)。

OLTP 主要用于银行业、航空、邮购订单、超级市场和制造业等的输入数据和取回交易数据。例如,银行为分布在各地的自动取款机(ATM)完成即时取款交易;机票预定系统每秒能处理的订票事务峰值可以达到 20 000 个。

OLTP 是事务处理从单机到网络环境的发展新阶段。OLTP 的特点在于事务处理量大,应用要求多个并行处理,事务处理内容比较简单且重复率高。大量的数据操作主要涉及的是一些增加、删除、修改、查询等操作。每次操作的数据量不大且多为当前的数据。

OLTP 处理的数据是高度结构化的,涉及的事务比较简单,数据访问路径是已知的,至少是固定的。事务处理应用程序可以直接使用具体的数据结构,如表、索引等。OLTP 数据库存储的数据量很大,经常每天要处理成千上万的事务,在处理业务数据时是非常有效的。

OLTP 面对的是事务处理操作人员和低层管理人员。但是,在为高层领导者提供决策分析时,则显得力不从心。

#### 2. 联机分析处理

关系数据库之父 E. F. Codd 在 1993 年提出,联机事务处理(OLTP)已经不能满足终端用户对数据库决策分析的需要,决策分析需要对多个关系数据库共同进行大量的综合计算才能得到结果。为此,他提出了多维数据库和多维分析的概念,即联机分析处理(On Line Analytical Processing,OLAP)概念。关系数据库是二维(平面)数据,多维数据库是空间立

体数据。

近年来,人们利用信息技术生产和搜集数据的能力大幅度提高,大量的数据库被用于商业管理、政府办公、科学研究和工程开发等,这一势头仍将持续发展下去。于是,一个新的挑战被提出来:在信息爆炸的时代,信息过量几乎成为人人需要面对的问题。如何才能不被信息的汪洋大海所淹没,从中及时发现有用的知识或者规律,提高信息利用率呢?要想使数据真正成为一个决策资源,必须充分利用它为一个组织的业务决策和战略发展服务才行,否则大量的数据可能成为包袱,甚至成为垃圾。OLAP是解决这类问题的最有力的工具之一。

OLAP专门用于支持复杂的分析操作,侧重对分析人员和高层管理人员的决策支持,可以应分析人员的要求快速、灵活地进行大数据量的复杂处理,并且以一种直观易懂的形式将查询结果提供给决策制定人,以便他们准确掌握企业(公司)的经营情况,了解市场需求,制定正确方案,增加效益。OLAP软件以它先进的分析功能和用多维形式提供数据的能力,正作为一种支持企业决策的解决方案而迅速崛起。

OLAP的基本思想是决策者从多方面和多角度,以多维的形式来观察企业的状态和了解企业的变化。

#### 3. OLTP与OLAP的对比

OLAP是以数据仓库为基础,其最终数据来源与OLTP一样均来自底层的数据库系统,但由于二者面对的用户不同,OLTP面对的是操作人员和低层管理人员,OLAP面对的是决策人员和高层管理人员,因而数据的特点与处理也明显不同。

OLTP和OLAP是两类不同的应用,它们各自的特点如表1.2所示。

**表1.2 OLTP与OLAP对比表**

| OLTP | OLAP |
|---|---|
| 数据库数据 | 数据仓库数据 |
| 细节性数据 | 综合性数据 |
| 当前数据 | 历史数据 |
| 经常更新 | 不更新,但周期性刷新 |
| 一次处理的数据量小 | 一次处理的数据量大 |
| 对响应时间要求高 | 响应时间合理 |
| 用户数量大 | 用户数量相对较小 |
| 面向操作人员,支持日常操作 | 面向决策人员,支持决策需要 |
| 面向应用,事务驱动 | 面向分析,分析驱动 |

### 1.1.3 数据字典与元数据

#### 1. 数据库的数据字典

数据字典是数据库中各类数据描述的集合,它在数据库设计中具有很重要的地位。数

据字典通常包括数据项、数据结构、数据流、数据存储和处理过程五个部分，其中数据项是数据的最小组成单位，若干个数据项可以组成一个数据结构，数据字典通过对数据项和数据结构的定义来描述数据流、数据存储的逻辑内容。

(1) 数据项

数据项是不可再分的数据单位。对数据项的描述通常包括数据项名、数据项含义说明、数据类型、长度、取值范围、取值含义等。

(2) 数据结构

数据结构反映了数据之间的组合关系。一个数据结构可以由若干个数据项组成，也可以由若干个数据结构组成。数据结构的描述通常包括数据结构名、含义说明、数据项等。

(3) 数据流

数据流是数据结构在系统内传输的路径，对数据流的描述通常包括数据流名、说明、数据流来源、数据流去向、平均流量等。其中“数据流来源”用于说明该数据流来自哪个过程。“数据流去向”用于说明该数据流将到哪个过程去。“平均流量”是指单位时间（如每天）里的传输次数。

(4) 数据存储

数据存储是数据结构保存数据的地方，数据存储的描述通常包括数据存储名、说明、编号、输入的数据流、输出的数据流、数据量、存取频度、存取方式。其中“存取频度”指每小时或每天或每周存取几次、每次存取多少数据等信息。“存取方式”包括是批处理还是联机处理、是检索还是更新、是顺序检索还是随机检索等。另外，“输入的数据流”要指出其来源，“输出的数据流”要指出其去向。

(5) 处理过程

处理过程一般用判定表或判定树来描述。数据字典中只需要描述处理过程的说明性信息，通常包括处理过程名、说明、输入、输出、处理。其中“处理”主要说明该处理过程的功能及处理要求。

可见，数据字典是关于数据库中数据的描述，而不是数据本身。数据字典是数据库的元数据。

**2. 数据仓库的元数据**

元数据（metadata）被定义为关于数据的数据（data about data）。元数据早期主要指网络资源的描述数据，用于网络信息资源的组织；其后，逐步扩大到各种以电子形式存在的信息资源的描述数据。目前，元数据这一术语实际用于各种类型信息资源的描述记录。

元数据在数据仓库中是描述数据仓库中数据及其环境的数据。数据仓库远比数据库复杂。在数据仓库中引入“元数据”的概念，它不仅仅是数据仓库的字典，而且还是数据仓库本身功能的说明数据。

元数据在数据仓库中不仅定义了数据仓库有什么，还指明了数据仓库中信息的内容和位置，刻画了数据的抽取和转换规则的说明，存储了与数据仓库主题有关的各种商业信息，而且整个数据仓库的运行都是基于元数据的，如数据的修改、跟踪、抽取、装入、综合以及使用等。由于元数据遍及数据仓库的所有方面，因此它已成为整个数据仓库的核心。

数据仓库的元数据共包含有四类元数据，除对数据仓库中数据的描述（数据仓库字典）外，还有以下三类元数据：

（1）关于数据源的元数据

数据仓库的数据源包含了很多不同数据库的数据结构，以及源数据的字段长度和数据类型。为数据仓库挑选数据时，必须将源数据的记录拆分，并将来自不同源数据的记录的某些部分组合起来。还要解决编码和字段长度不同的问题。当将这些信息传递给最终数据仓库的时候，必须把这些数据与原始数据联系起来。

（2）关于抽取和转换的元数据

这类元数据包含了源数据系统的数据抽取方法、数据抽取规则以及抽取频率等数据转换的所有说明数据。

（3）关于最终用户使用数据仓库的元数据

最终用户使用数据仓库的元数据是数据仓库的导航图。它使最终用户可以从数据仓库中找到自己需要的信息。

### 1.1.4 数据仓库的定义与特点

数据仓库（Data Warehouse）的概念是由 W. H. Inmon 在《建立数据仓库（Building the Data Warehouse）》一书中提出的。数据仓库的提出是以关系数据库、并行处理和分布式技术为基础的信息新技术。

从目前的发展形势看，数据仓库技术已紧跟 Internet 而上，成为信息社会中获得企业竞争优势的又一关键技术。

#### 1. 数据仓库的定义

（1）W. H. Inmon 对数据仓库的定义

数据仓库是面向主题的、集成的、稳定的、不同时间的数据集合，用于支持经营管理中决策制定过程。

（2）SAS 软件研究所的观点

数据仓库是一种管理技术，旨在通过通畅、合理、全面的信息管理，达到有效的决策支持。

从数据仓库的定义可以看出，数据仓库是明确为决策支持服务的，而数据库是为事务处理服务的。

#### 2. 数据仓库的特点

从数据仓库的定义可以看出数据仓库有如下特点。

（1）数据仓库是面向主题的

主题是数据归类的标准，每一个主题基本对应一个宏观的分析领域。例如，保险公司的数据仓库的主题为客户、政策、保险金、索赔等。

基于应用的数据库组织则完全不同，它的数据只是为处理具体应用而组织在一起的。保险公司按应用组织的数据库是汽车保险、生命保险、健康保险、伤亡保险等。

(2) 数据仓库是集成的

数据进入数据仓库之前,必须经过加工与集成。对不同的数据来源进行统一数据结构和编码。统一原始数据中的所有矛盾之处,如字段的同名异义、异名同义、单位不统一、字长不一致等。总之,将原始数据结构做一个从面向应用到面向主题的大转变。

(3) 数据仓库是稳定的

数据仓库中包括了大量的历史数据。数据经集成进入数据仓库后是极少或根本不更新的。

(4) 数据仓库是随时间变化的

数据仓库内的数据时限在 5～10 年,故数据的键码包含时间项,标明数据的历史时期,这适合决策分析时进行时间趋势分析。

而数据库只包含当前数据,即存储某一时间的正确的有效数据。

(5) 数据仓库中的数据量很大

通常的数据仓库的数据量为 10GB 级,相当于一般数据库(约 100MB)的 100 倍,大型数据仓库是 1TB(1000GB)级数据量。

数据仓库中数据量的比重是索引和综合数据占 2/3,原始数据占 1/3。

(6) 数据仓库软硬件要求较高

① 需要一个巨大的硬件平台;

② 需要一个并行的数据库系统。

## 1.2 数据挖掘的兴起

### 1.2.1 从机器学习到数据挖掘

数据挖掘来源于机器学习。学习是人类具有的智能行为,主要目的在于获取知识。机器学习是研究使计算机模拟或实现人类的学习行为,即让计算机通过算法自动获取知识。机器学习是人工智能领域中的重要研究方向。

20 世纪 60 年代开始了机器学习的研究。比较典型的成果有:Rosenblate 的感知机,它是最早用神经网络进行模式识别的方法;Sammel 的西洋跳棋程序,他用线性表达式的启发式方法,通过多次人机对弈,自动修改表达式中的系数,使程序逐渐聪明,该程序竟然取得了胜过作者和州冠军的成绩。

20 世纪 80 年代,机器学习取得了较大的成果。Michelski 等人的 AQ11 系统(1980),能从大量病例中归纳出大豆病症的判断规则。AQ11 是一个很成功的归纳学习方法;Quiulan 的 ID3(1983)决策树方法影响很大,实用性很强;Langley 等人的 BACON 系统(1987)能重新发现物理学的大量规律;Rumelhart 等人研制的反向传播神经网络 BP 模型(1985)为神经网络的学习开创了一个新阶段。

这些显著成果的出现,使"机器学习"逐渐形成了人工智能的主要学科方向之一。1980 年在美国召开了第一届国际机器学习学会研讨会;1984 年《机器学习》杂志问世。

我国在 1987 年召开了第一届全国机器学习研讨会。1989 年成立了中国人工智能学会

机器学习学会。我国学者洪家荣研制的AE1系统(1985)采用了扩张矩阵方法；钟鸣和本书作者研制的IBLE方法(1992)利用信道容量建立决策规则树，识别效果比ID3方法更高。本书作者研制的FDD经验公式发现系统(1998)，能发现含初等函数或复合函数的经验公式，发现的公式比BACON系统发现的公式范围更宽。

1989年在美国召开了第一届知识发现(KDD)国际学术会议，从数据库中发现知识(Knowledge Discovery in Database，KDD)形成了新概念。KDD研究的问题有：①定性知识和定量知识的发现；②知识发现方法；③知识发现的应用等。

1995年在加拿大召开了第一届知识发现(KDD)和数据挖掘(DM)国际学术会议。由于把数据库中的“数据”形象地比喻成矿床，因此“数据挖掘(Data Mining，DM)”一词很快流传开来。

数据挖掘是知识发现中的核心工作，主要研究发现知识的各种方法和技术。而这些方法和技术主要来自于机器学习。随着数据挖掘的发展，出现了一些新的数据挖掘方法，如大型数据库库中关联规则的挖掘，利用粗糙集进行属性约简和规则获取等。

数据挖掘兴起时主要是在数据库中挖掘知识，随着数据仓库的出现和发展，很快将数据挖掘技术和方法用于数据仓库。典型的啤酒与尿布的故事(这两种商品同时出售出现的概率很大)就是在数据仓库中挖掘出的关联知识。

### 1.2.2 数据挖掘含义

按《人工智能辞典》的定义：信息是数据中所蕴涵的意义。知识是人们对客观世界的规律性认识。

数据库中每个数据记录的内含代表了该记录的信息。而数据挖掘是从数据库中所有数据记录中归纳总结出知识。知识的数量大大少于数据记录量。这些知识代表了数据库中数据信息的规律，即用少量的知识能够覆盖数据库中所有的记录。

例如，人口数据库中存储各国人口的记录，它将是一个庞大的数据库。但是，通过数据挖掘，可以得出形式化表示的规则知识：

$$(头发=黑色)\vee(眼睛=黑色)\rightarrow 亚洲人$$

其中“∨”表示“或”，“→”表示“蕴涵”，规则知识表示为“若(条件)则(结论)”，即表示若头发是黑色或者眼睛是黑色的人，则他是亚洲人。

该知识代表了亚洲人的特点，也即覆盖了所有亚洲人的记录。

知识的获得是通过数据挖掘算法，如AQ11方法、ID3方法等经过计算得到的。

### 1.2.3 数据挖掘与OLAP的比较

#### 1. OLAP的多维分析

OLAP是在多维数据结构上进行数据分析的。同时对多维数据进行分析是复杂的。一般在多维数据中取出(切片、切块)二维或三维数据来进行分析，或对层次的维进行钻取操作，向下钻取获得更详细的数据，向上钻取获得更综合的数据。

OLAP要适应大量用户同时使用同一批数据，适应于不同地理位置的分散化的决策。

OLAP 的功能和算法包括聚合、分配、比率、乘积等描述性的建模功能。

OLAP 平时需要查询大量的日常商业活动信息，如每周的布匹购买量、每周布匹的内部库存以及布匹的销售量等。OLAP 更需要查询商业活动的变化情况，如每周布匹购买量的变化值、衣服生产量的变化值、衣服销售价格的变化等。这些变化值对经理们制定决策更重要。

经理们往往从查询出的变化值中，通过 OLAP 追踪查询找出存在的原因。例如，经理看到利润小于预计值的时候，他可能会深入到各个国家查看整个产品利润情况。这样，他可能发现有些国家的利润明显低于其他国家，于是他自然就会查看这些国家中不同产品组的利润情况，总的目标就是寻找一些比较异常的数据来解释某种现象。经过一番观察之后，就会发现非直接成本在这些国家明显偏高。进一步对这些非直接成本进行分析，可以发现近期对于某些产品的赋税明显增加，从而明显影响了最终的利润。这种分析查询要求时间响应快。

以上是 OLAP 的典型应用，通过商业活动变化的查询发现的问题，经过追踪查询找出问题出现的原因，达到辅助决策的作用。

#### 2. 数据挖掘

OLAP 是在带层次的维度和跨维度进行多维数据分析的。数据挖掘则不同，它是以变量和记录为基础进行分析的。

数据挖掘任务在于聚类（如神经网络聚类）、分类（如决策树分类）、预测等。这些是带有探索性的建模功能。

数据挖掘在于寻找不平常的且有用的商业运作模型。考察数据的不同类型或者找出变量之间的关系。数据挖掘需要察看海量数据，主要是详细数据和历史数据。为此经常将数据仓库中的数据拷贝到一个专门的存储器上，对数据的挖掘分析可能要花去大量的时间，即不要求快速分析。数据挖掘人员有时并不能精确地知道什么是必须分析的，有时数据挖掘一无所获。但是，有时通过数据挖掘会发现意外的、无价的信息“金块”。例如，如果能够确定一个高价值的客户或可能离开的客户特征，就可以要求公司采取措施保留这些客户，这比从竞争对手那里重新争取曾经失去的客户所需的费用少得多。

### 1.2.4 数据挖掘与统计学

#### 1. 统计学的发展过程

统计学是一门有悠久历史的学科。统计学开始于十七世纪，它与国家政治有紧密的关系。英国人 W. Petty(1623—1682)的《政治算术》一书中第一次用计量和比较的方法，对英国与法、意、荷等国进行国力比较。J. Graunt(1620—1674)通过统计计算，发现男女人数占人口数的比例大致相同、出生儿中男婴比例稍高、婴幼儿的死亡率较大等规律性的现象。

17 世纪，B. Pascal 等人提出“概率”概念，用来描述某一事件发生的可能性。18 世纪，在观测天体运动时会有误差产生，虽然多次测量，由于有误差，得到的总是和真值不同的值。高斯(Gauss，1777—1855)提出误差值落在$(a,b)$区间的概率等于该区间上正态分布曲线下

的面积，称误差服从正态分布（高斯分布）。比利时的凯特勒（A. Quetelet，1796—1874）称“支配着社会现象的法则和方法是概率论”。

近代统计学重视社会调查。通过对全部对象（总体）进行调查，为制定计划和决策提供依据，如果对总体的某些分布情况有一定把握的话，就不必搞全面调查，可以搞部分调查，即抽样调查，由部分推断全部。概率论和数理统计理论起着重要的作用。现在，各国在进行经济统计、国事调查、社会调查、收视率调查、民意测验时，采用的几乎都是抽样调查。

现代统计学，从线性到非线性、从低维到高维、从显在到潜在、从连续到离散等方面有较完备的理论和方法。统计软件包 SPSS、SAS 等已经普及，统计工作基本上利用计算机来完成。

**2. 统计学中应用于数据挖掘的内容**

（1）常用统计

在大量数据中求最大值、最小值、总和、平均值等。

（2）相关分析

通过求变量间的相关系数来确定变量间的相关程度。

（3）回归分析

建立回归方程（线性或非线性）以表示变量间的数量关系，再利用回归方程进行预测。

（4）假设检验

在总体存在某些不确定情况时，为了推断总体的某些性质，提出关于总体的某些假设，对此假设利用置信区间来检验，即任何落在置信区间之外的假设判断为“拒绝”，任何落在置信区间之内的假设判断为“接受”。

（5）聚类分析

对样品或变量进行聚类的方法。具体方法是把每一个样品看成是 $m$ 维空间的一个点，聚类是把“距离”较近的一些点归为同一类，而将“距离”较远的点归为不同的类。

（6）判别分析

建立一个或多个判别函数，并确定一个判别标准。对未知对象利用判别函数将它划归某一个类别。

（7）主成分分析

主成分分析是把多个变量化为少数的几个综合变量，而这几个综合变量可以反映原来多个变量的大部分信息。

主成分分析的一种推广是因子分析，即用少数几个因子（$F_i$），去描述许多变量（$X_j$）之间的关系。变量（$X_j$）是可以观测的显在变量，而因子（$F_i$）是不可观测的潜在变量。

**3. 统计学与数据挖掘的比较**

统计学主要是对数量数据（数值）或连续值数据（如年龄、工资等）进行数值计算（如初等运算）的定量分析，得到数量信息，如常用统计量（最大值、最小值、平均值、总和等），相关系数、回归方程等。

数据挖掘主要对离散数据（如职称、病症等）进行定性分析（覆盖、归纳等），得到规则知

识。例如,如果某人的眼睛是黑的或者头发是黑的,则可以认为他是亚洲人。

在统计学中有聚类分析和判别分析,它们与数据挖掘中的聚类和分类相似,但是采用的标准不一样。统计学的聚类采用的“距离”是欧式距离,即两点间的坐标(数值)距离;而数据挖掘的聚类采用的“距离”是汉明距离,即属性取值是否相同,相同者距离为“0”,不相同者距离为“1”。

总之,统计学与数据挖掘是有区别的,但是它们之间是相互补充的。不少数据挖掘的著作中均把统计学的不少方法引入到数据挖掘中,与将机器学习中的不少方法引入到数据挖掘中一样,作为从数据获取知识的一大类方法。

虽然统计学的不少方法可以归入到数据挖掘中,但统计学仍然是一门独立的学科。

## 1.3 数据仓库和数据挖掘的结合

### 1.3.1 数据仓库和数据挖掘的区别与联系

#### 1. 数据仓库与数据挖掘的区别

数据仓库是在数据库的基础上发展起来的。它将大量的数据库的数据按决策需求进行重新组织,以数据仓库的形式进行存储,它将为用户提供辅助决策的随机查询、综合信息以及随时间变化的趋势分析信息等。

数据仓库是一种存储技术,它的数据存储量是一般数据库的100倍,它包含大量的历史数据、当前的详细数据以及综合数据。它能适应于不同用户对不同决策需要提供所需的数据和信息。

数据挖掘是从人工智能机器学习中发展起来的。它研究各种方法和技术,从大量的数据中挖掘出有用的信息和知识。最常用的数据挖掘方法是统计分析方法、神经网络方法和机器学习中研究的方法。数据挖掘中采用机器学习的方法有归纳学习方法(覆盖正例排斥反例方法,如AQ系列算法、决策树方法、关联规则挖掘等)、遗传算法、发现学习算法(如公式发现系统BACON)等。

利用数据挖掘的方法和技术从数据仓库中挖掘的信息和知识,反映了数据仓库中数据的规律性。用户利用这些信息和知识来指导和帮助决策。例如,利用分类规则来预测未知实体的类别。

#### 2. 数据仓库与数据挖掘的关系

数据仓库与数据挖掘都是决策支持新技术。但它们有着完全不同的辅助决策方式。数据仓库中存储着大量辅助决策的数据,它为不同的用户随时提供各种辅助决策的随机查询、综合信息或趋势分析信息。数据挖掘是利用一系列算法挖掘数据中隐含的信息和知识,让用户在进行决策中使用。

数据仓库和数据挖掘可以结合起来。在数据仓库系统前端的分析工具中,数据挖掘是其中重要的工具之一。它可以帮助决策用户挖掘数据仓库的数据中隐含的规律性。

数据挖掘用于数据仓库实现决策支持,具体表现为:

（1）预测客户的购买倾向；
（2）进行客户利润贡献度分析；
（3）分析欺诈行为；
（4）进行销售渠道优化分析等。
数据仓库和数据挖掘的结合对支持决策会起到更大的作用。

**3. 数据仓库中数据存储特点**

数据挖掘兴起是针对数据库的，随着数据仓库的兴起和发展，由于数据仓库不同于数据库，数据挖掘也随之发生了变化。

（1）数据存储方式的不同

数据库的数据存储是按照管理业务中事物处理项目的要求而存放的。

数据仓库的数据存储是按决策分析需求而存放的。这种需求是以决策主题为对象的，典型的主题是客户。这样，在数据仓库中，客户数据需要从多个数据库集成而来，如银行数据仓库需要从储蓄、信用卡、贷款等不同数据库中，对同一客户的数据进行抽取并集成在一起，以便完成对该客户的分析。

（2）数据存储的数据量的不同

数据库的数据存储量相对数据仓库的数据存储量小得多。从上面的例子可以看出，以客户主题建立数据仓库的数据量是储蓄、信用卡、贷款三个数据库的数据量的总和。按一般的统计，数据仓库的数据量是数据库数据量的 100 倍。数据仓库的数据量比数据库的数据量大这么多，原因在于：①数据仓库中的数据（近期基本数据）是数据库中数据按决策主题重新组织并集成而来的；②数据仓库中的数据还需要保留大量的历史数据，用于预测分析；③数据仓库为了给不同级别管理者提供各种决策分析的数据，需要对近期基本数据进行轻度综合和高度综合，这些综合数据在数据仓库中占据了不小的比重。近期基本数据、历史数据、综合数据三者的数据相加，使数据仓库的数据量远远大于数据库中的数据量。

（3）数据存储的结构不同

由于数据仓库的数据量远大于数据库的存储量，因此数据库的关系型二维（平面）存储格式不能适应于数据仓库。数据仓库的数据存储结构采用多维的超立方体结构形式。数据仓库的数据存储结构采用星型模型或者多维立体数据库形式。

**4. 数据仓库中数据挖掘特点**

数据仓库的最大应用在于扩展市场，制定营销策略，争取更多的客户。

（1）数据挖掘从数据仓库中挖掘的信息

数据挖掘应用于数据仓库后，能挖掘更深层次的信息，如：

① 哪些商品一起销售好？（利用关联分析）
② 偏爱某类商品的客户特征是什么？（利用聚类和分类分析）
③ 还有哪些客户具有上述特征？（利用类比分析）
④ 哪些商业事务处理可能有欺诈性？（利用神经网络）
⑤ 高价值客户的共同点是什么？（利用分类分析）

典型的例子是通过数据挖掘对高价值客户以及可能离开的客户进行挖掘,得出它们的特征,这样就让公司做出决策,达到保留这些高价值的客户和争取可能离开的客户,从而提高公司的利润。

(2) 数据仓库对数据挖掘提出了新要求

① 数据挖掘需要可扩展性:

数据挖掘对数据仓库的应用一般使用的数据是详细数据,不用综合数据,因为综合数据“平滑”了数据间的差别,从而无法发现单个数据项目之间的微妙相关性。

数据仓库中的数据随着时间的推移逐渐增长。这样,数据挖掘方法就应该具有可扩展性,能够处理递增的数据量。

② 数据挖掘方法需要能挖掘多维知识:

数据仓库中的数据模型是多维数据组织,它不同于数据库的二维数据组织。数据挖掘应用到数据仓库时需要能挖掘多维数据知识。

例如,对数据库的关联分析只能得到同一个商品维中不同商品之间的关联关系。到数据仓库中的关联分析就应该能对多维数据寻找它们的关联关系,即除不同商品的关联外,还要找出商品与商店或时间等不同维之间的关联关系。

### 1.3.2 基于数据仓库的决策支持系统

在建立数据仓库之前,利用数据库来完成决策分析时,由于决策者不能明确表明他到底需要哪些具体数据来帮助辅助决策,一开始会提出一个粗糙的需求,由 IT(信息技术)人员编写专门程序从数据库中抽取数据,形成所需的报告。决策者根据这个报告会马上想起需要更多的数据,提供新的报告。IT 人员重新编写程序抽取新的数据,完成新的报告。

由于决策的不明确性,对数据抽取的多样性,包括不同时间的抽取以及不同角度的抽取,形成的分析报告会造成不同的结果,甚至矛盾的结果。例如,一个 IT 人员提出的分析报告说企业的业绩下降了 15%,另一个 IT 人员提出的分析报告说企业的业绩上升了 10%。这两个结论不但不吻合,而且相去甚远。这让决策者很难相信报告结论的正确性,也无法帮助他做出决策。

从而认识到在数据库的基础上编写专门的程序,获取信息辅助决策是不成功的。人们把用这种方式建立的决策支持系统认为是失败的。

为了建立随时提取销售量最好的产品名单,告诉出现问题的地区,并能分析出现问题的原因,对比各种数据,显示最大的利润等辅助决策信息的决策支持系统,数据仓库成了唯一可行的解决方案。

数据仓库对整个企业各部门的数据进行统一和综合,这实际上是对决策支持的一次革新。企业可以用它来取得各个重要方面的数据与分析结果,例如商品利润、市场分析和风险管理等,从而改善企业的自身管理。举例来说,数据仓库用户可以立即得到其单位当前所处地位的准确报告,了解其公司面临的风险,包括各项事务及整个企业所有业务面临的风险,并对市场和法规条例的需要迅速做出反应。

数据仓库的决策支持功能有:

(1) 对当前和历史数据完成查询和报表处理;

(2) 可以用不同方法进行“如果，将怎样(what-if)”分析；

(3) 可以查询细节，查询综合，并能深入追踪查询(钻取分析)；

(4) 认清过去的发展趋势，并将其应用于对未来结果的分析。

数据仓库是为辅助决策而建立的，单依靠数据仓库达到辅助决策的能力是有限的。数据仓库中有大量的综合数据，这些数据为决策者提供了综合信息，即反映企业或部门的宏观状况。数据仓库保存有大量历史数据，这些数据通过预测模型计算可以得到预测信息。

综合信息与预测信息是数据仓库所获得的辅助决策信息。

数据仓库(DW)中增加联机分析处理(OLAP)和数据挖掘(DM)等分析工具，能较大地提高辅助决策能力。联机分析处理(OLAP)对数据仓库中的数据进行多维数据分析，即多维数据的切片、切块、旋转、钻取等，只有通过分析更详细的数据，才能得到更深层中的信息和知识。例如节假日销售的影响、某日的促销活动的影响等，这些信息在综合数据中是反映不出来的。数据挖掘(DM)技术能获取关联知识、时序知识、聚类知识、分类知识等。只有通过数据挖掘技术对数据仓库中数据的挖掘，才能获取更多的辅助决策信息和知识。

数据仓库(DW)和联机分析处理(OLAP)及数据挖掘(DM)相结合的决策支持系统，是以数据仓库为基础的，被称为基于数据仓库的决策支持系统，其结构如图1.1所示。

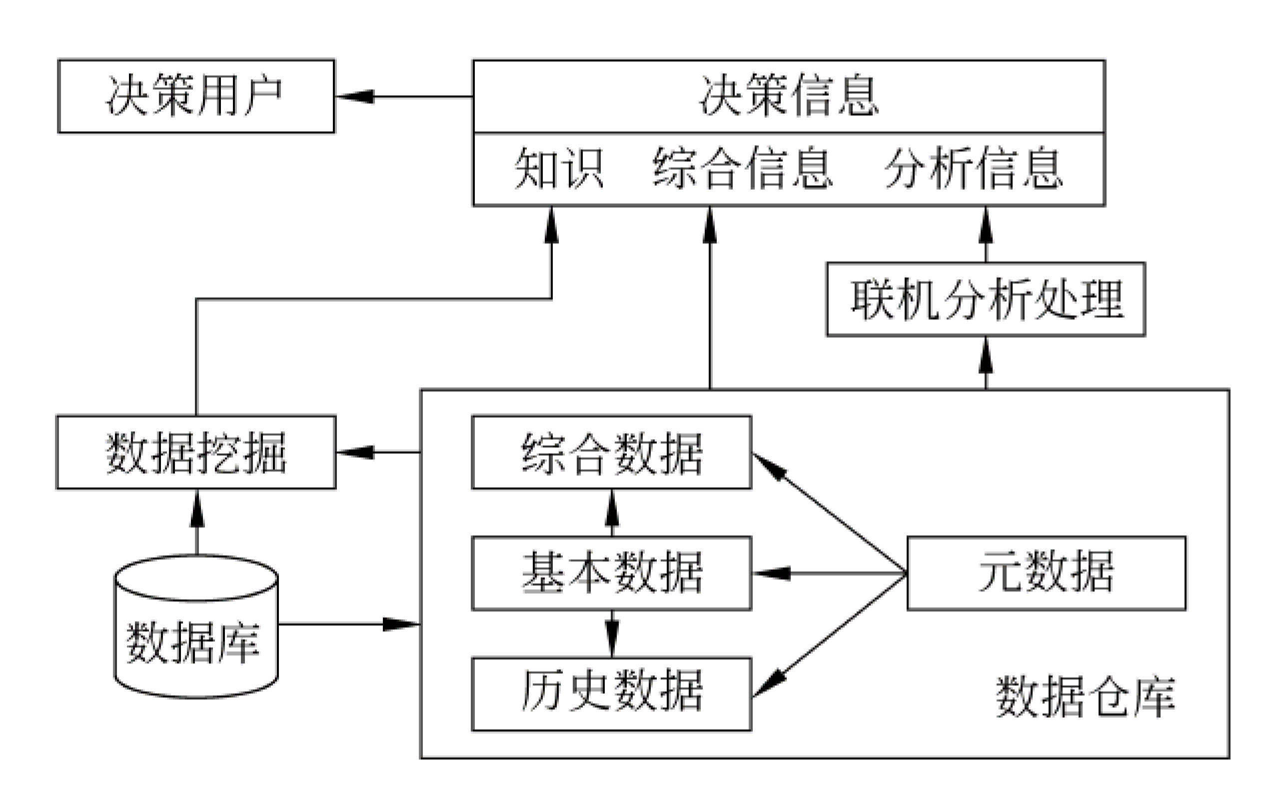



图1.1 基于数据仓库的决策支持系统结构

概括地说，基于数据仓库的决策支持系统是从数据仓库的数据中获取辅助决策的信息和知识，为决策提供支持。

基于数据仓库的决策支持系统不同于20世纪80年代出现的基于模型的决策支持系统和20世纪90年代兴起的智能决策支持系统。因此，把基于模型和知识的智能决策支持系统称为传统的决策支持系统，而把基于数据仓库的决策支持系统称为新决策支持系统。

### 1.3.3 数据仓库与商业智能

#### 1. 商业智能的概念

商业智能是在20世纪90年代中期提出的。商业智能以数据仓库为基础，通过联机分析处理和数据挖掘技术帮助企业领导者针对市场变化的环境，做出快速、准确的决策。

商业智能与新决策支持系统从组成和目标来看是一致的。但是，商业智能是一种技术，新决策支持系统是解决实际决策问题的一个系统。可以理解为：新决策支持系统是利用商

业智能技术来解决实际决策问题的系统。

数据仓库、联机分析处理与数据挖掘组成的商业智能所体现的智能行为在于,能够解决市场环境中随机变化的决策问题。由于市场千变万化,每次需要解决的决策问题都不相同。解决随机出现的问题需要利用智能的手段。商业智能所提供的智能手段表现为联机分析处理的任意切片、切块和钻取,以及利用数据挖掘技术所获得的知识。

**2. 商业智能辅助制定更好更快的决策**

公司需要制定的决策有两类:由高层管理者制定宏观的战略决策;基层人员在日常事务中制定决策。战略决策有:投资哪个项目?哪些业务需要分离还是合并?制定销售策略等。事务决策有:销售员决定是否给一个客户折扣;生产经理决定是否投产一个新产品以满足客户需求;市场营销专家决定是否要进行新一轮的直接邮购活动;采购经理决定是否买更多的材料;等等。这些事务决策只具有“战术”意义,不会影响到业务运作的基础,但从总体效果看,其重要性并不亚于企业高级管理人员做出的重大决策,也会直接影响企业的成败。这些决策很少是通过决策分析做出的,大多靠的是经验、积累的知识和惯常的做法。提高企业日常工作中的决策质量,将直接对企业的成本和营业收入产生影响。

商业智能改进企业决策过程,表现在如下方面。

(1) 信息共享

有了商业智能系统就可以实现信息共享,用户可以迅速找到所需要的数据,通过对数据进行钻取分析以达到目标。例如,某公司通过商业智能系统跟踪商品的质量管理,能及时发现问题,而不是一个星期后再查阅各种报告来发现问题。时间的节省以及产品质量的提高,不仅降低了企业的成本,也给公司带来了更多的收入。

(2) 实时反馈分析

商业智能的运用能够使员工随时看到工作进展程度,并且了解一个特定的行为对现实目标的效用。如果员工们都能看到自己的行为如何提升或者影响了业绩,那么也就不需要过于复杂的激励体系了。

例如朋斯卡物流公司,司机的激励机制与其驾驶表现,如每英里的耗油量和损耗程度等成本控制方面的因素相关联。通过电子商业智能系统,公司的主控电脑就能根据司机出车行驶的里程计算出每加仑汽油能支持的里程数,然后再把数据传输到数据仓库,通过数据仓库,员工们就可以分析提高绩效的可能性,即发现如何调整汽车保养或司机驾驶习惯来达到业绩目标,提高业务水平并创造更多的价值。

(3) 鼓励用户找出问题的根本原因

根据初步得到的答案而采取的行动可能未必正确,因为初步的探究往往没有发现根本问题的所在。要找出根本原因就需要对与成功或失败相关的诸多因素进行深度分析。

通过企业商业智能系统,能够找到某部门业绩糟糕或者出色的根本原因,只要不断地追问“为什么?为什么?”。这个过程可能是从分析一个报告开始,比如每季度的销售情况,每个答案引出一个新问题,采取钻取或分析方法,就能把最根本的原因找出来。例如,通过企业商业智能系统,制衣商发现他们推出的市场促销活动效果不理想。在分析了诸多数据后,

制衣商开始把价格与市场需求进行灵活挂钩。结果,该制衣商缩短了存货时间,提高了存货管理的效率,营运资本、销售、利润等几项主要业绩指标也明显好转。

(4) 使用主动智能

在数据仓库中设定预警机制,一旦出现超过预警条件的数据,就自动通过各种设备,比如电子邮件、传呼、手机等通知用户。这种主动智能有助于用户及时决断,并采取相应措施。

(5) 实时智能

企业采用真正的实时智能,将大大提高运营效率、降低成本、提高服务质量。例如,朋斯卡物流公司认识到需要一个商业智能系统来实时监控和智能管理运输和物流业务,该系统掌握了很多信息,把货物运载量维持在一个最高的水平,帮助客户更快地把货物从A地送到B地。企业商业智能系统能实时跟踪卡车的货物装载量,如果一辆卡车的装载量只有最大装载量的一半,公司根据商业智能系统发出指令让该车调整路线,再装载一些货物。该系统使公司的所有营业收入上升了很多。

# 习 题 1

1. 数据库与数据仓库的本质差别是什么?
2. 从数据库发展到数据仓库的原因是什么?
3. 举例说明数据库与数据仓库的不同。
4. 说明OLTP概念和OLAP概念。
5. OLTP如何在网络数据库上进行事务处理?
6. 说明OLTP与OLAP的主要区别。
7. 数据库中数据字典包括哪些内容?
8. 元数据的定义是什么?
9. 元数据与数据字典的关系是什么?
10. 数据仓库的定义是什么?
11. 数据仓库的特点有哪些?
12. 说明机器学习如何形成人工智能的学科方向。
13. 说明数据挖掘的含义。
14. OLAP多维分析如何辅助决策?举例说明。
15. 数据挖掘与OLAP有什么不同?
16. 举例说明统计学的价值。
17. 说明统计学应用于数据挖掘中所包含的内容。
18. 说明统计学与数据挖掘的不同。
19. 说明数据仓库与数据挖掘的区别与联系。
20. 数据挖掘应用于数据库与数据挖掘应用于数据仓库有什么不同?
21. 举例说明数据挖掘从数据仓库中挖掘的信息有哪些。
22. 数据仓库对数据挖掘提出了哪些新要求?

23. 数据仓库与联机分析处理、数据挖掘在决策支持方面有什么不同？
24. 基于数据仓库的决策支持系统的组成是什么？
25. 画出基于数据仓库的决策支持系统结构图。
26. 说明基于数据仓库的决策支持系统与传统决策支持系统有什么区别。
27. 商业智能概念是什么？
28. 如何理解商业智能与基于数据仓库的决策支持系统的区别和联系？
29. 商业智能在哪些方面改进企业决策过程？

# 第2章　数据仓库原理

## 2.1　数据仓库结构体系

### 2.1.1　数据仓库结构

数据仓库是在原有关系型数据库基础上发展形成的，但不同于数据库系统的组织结构形式。它从原有的大量业务数据库中获得的数据，经过转换后形成当前基本数据层；它经过综合后形成轻度综合数据层；轻度综合数据再经过综合后形成高度综合数据层。W. H. Inmon 在《建立数据仓库》一书中给出数据仓库的结构如图 2.1 所示。数据仓库结构包括当前基本数据（current detail data）、历史基本数据（older detail data）、轻度综合数据（lightly summarized data）、高度综合数据（highly summarized data）、元数据（meta data）。

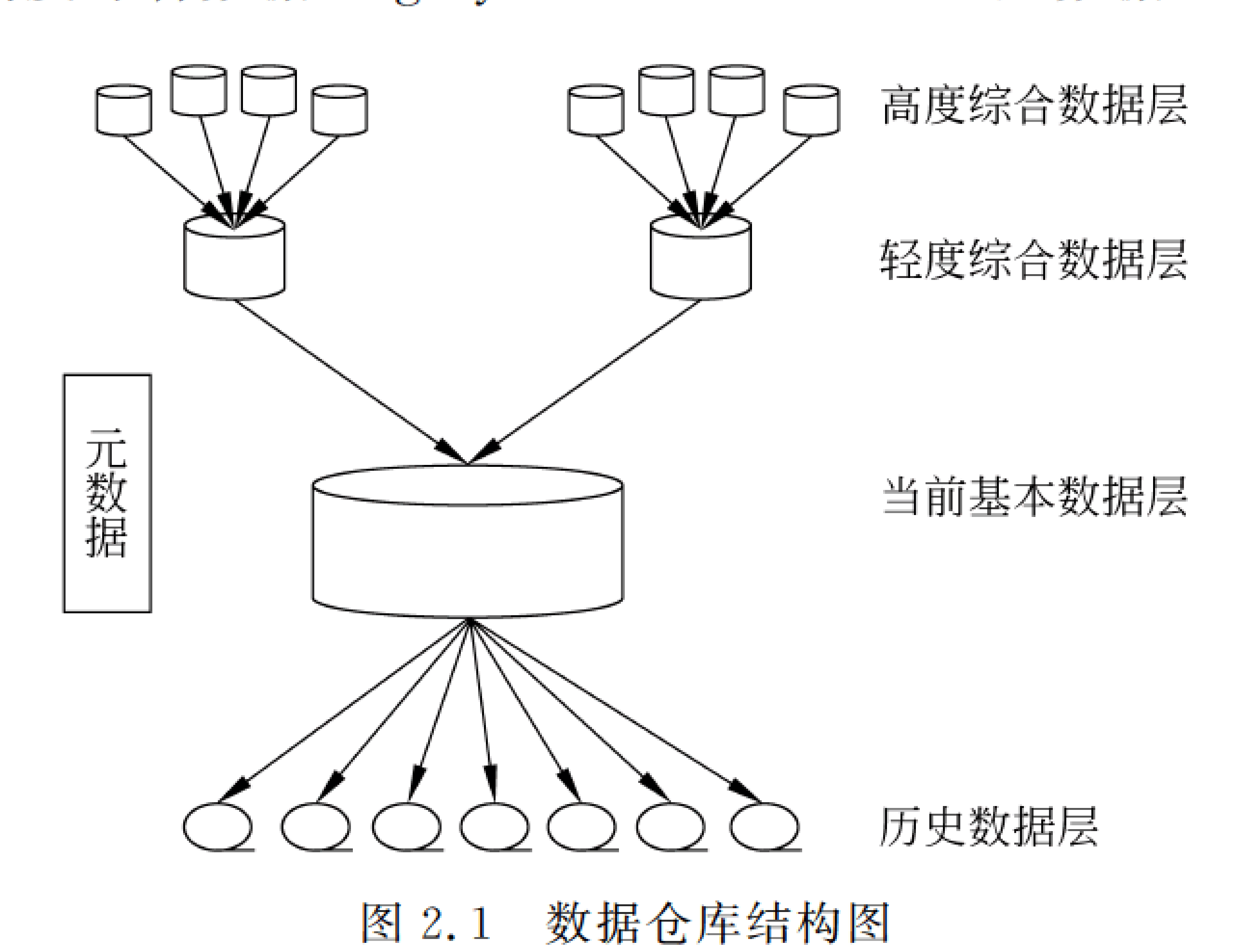



图 2.1　数据仓库结构图

当前基本数据是最近时期的业务数据，是数据仓库用户最感兴趣的部分，数据量大。当前基本数据随时间的推移，由数据仓库的时间控制机制转为历史基本数据，一般被转存于介质中，如磁带等。轻度综合数据是从当前基本数据中提取出来的，设计这层数据结构时会遇到"综合处理数据的时间段选取，综合数据包含哪些数据属性（attributes）和内容（contents）"等问题。最高一层是高度综合数据层，这一层的数据十分精练，是一种准决策数据。

整个数据仓库的组织结构是由元数据来组织的，它不包含任何业务数据库中的实际数据信息。元数据在数据仓库中扮演了重要的角色，它包括如下信息：①数据仓库的目录信息（数据字典）；②数据从数据库环境向数据仓库环境转换时对应的说明；③指导从当前基本数据到综合数据的综合方式的说明；④指导用户使用数据仓库。

在数据库中只存储当前的详细数据。而数据仓库除存储按主题组织起来的当前详细数据外，还需要存储综合数据，这是为适应决策需求而增加的。在数据库中需要得到综合数据

时，采用数据立方体方法（见 3.4.4 节）对详细数据进行综合。在数据仓库中并不采取临时计算的方式得到综合数据，而是在用户提出需要综合数据之前，就预先将可能需要的综合数据利用数据立方体计算好，存入综合数据层中，这种综合数据层在用户查询时，能迅速提供给用户。为此，在建数据仓库时，要分析好各类用户可能需要哪些综合数据，并将这些综合数据都存储在综合数据层中。

综合数据与详细数据是不同“粒度”的数据。粒度是指数据仓库的数据单元中保存数据的细化或综合程度的级别。细化程度越详细，粒度级就越低。

不同粒度数据的存储数据量差距很大。例如，在低粒度级（详细数据）上，每次电话都详细记录下来，一个月每位顾客平均有 200 条记录，总共需要 40 000 个字节。在高粒度级（综合数据），每位顾客只有一个记录，大约只需要 200 个字节。

高粒度级不仅只需要少得多的字节存放数据，而且只需要较少的索引项。这样的数据存储效率较高。

在数据仓库环境中，粒度之所以是设计数据仓库的一个重要方面，不仅因为它影响了存放在数据仓库中的数据量的大小，它同时也影响数据仓库所能回答的查询类型。当提高数据粒度时（综合数据），数据所能回答查询的能力将会随之降低。而很小粒度的数据（详细数据）可以回答任何问题，但在高粒度的数据上（综合数据），可以回答的问题具有宏观性。

例如，提出如下查询：“张三上星期是否给他在外地的女友打了电话？”。在低粒度级上这个问题是可以回答的，这需要查阅大量的记录，该查询最终总是可以确定的。然而在高粒度级上就无法明确回答这个问题，因为在高粒度级上只存放有张三打出电话的总数，并不能确定其中是否有一个电话是打往外地女友的。

但是，在进行决策分析时，很少对单个事件进行查询，通常是针对某个数据集合进行处理的（这在数据仓库环境中是常见的）。例如，提出综合查询：“上个月人们从广州打出的长途电话平均有多少个？”。在决策分析中，这种类型的查询非常多。该查询既可以在高粒度上也可以在低粒度上进行处理。但在回答这个问题时，在不同粒度级上所使用的资源具有相当大的差别。在低粒度级上回答这个问题需要查询每一个记录，使用大量的资源来回答这个问题。在高粒度级上，包括了足够的细节（如包括每个顾客打出长途电话的次数），则使用高粒度级数据的效率就会高很多。例如，在轻度综合级上电话记录如下，将能使用较少的资源回答以上问题：

三月份，李四，电话数量：46 个；电话平均长度：10 分钟；长途电话数：12 个；等等。

在数据仓库中存储多种粒度数据（详细层、轻度综合层、高度综合层等）是为了提高决策分析效果。大部分决策分析处理是针对存储效率高的轻度综合层数据进行的。当需要分析更低的细节级数据（占 5%或者更少的可能）时，可以到详细数据层数据上进行。在详细数据层上访问数据是昂贵的、复杂的。

### 2.1.2 数据集市及其结构

数据仓库是企业级的，能为整个企业各个部门的运行提供决策支持手段；而数据集市则是部门级的，一般只能为某个局部范围内的管理人员服务，因此也称之为部门级数据仓库（Departmental Data Warehouse）。

### 1. 数据集市的产生

数据仓库的工作范围和成本常常是巨大的。信息技术部门必须对所有的用户并以全企业的眼光对待任何一次决策分析。这样,就形成了代价很高的、耗时较长的大项目。

于是提供更紧密集成的,拥有完整图形接口并且价格吸引人的工具——数据集市(Data Marts),就应运产生。

目前,全世界对数据仓库总投资的一半以上均集中在数据集市上。

### 2. 数据集市的概念

数据集市是一种更小、更集中的数据仓库,为公司提供了一条分析商业数据的廉价途径。

数据集市是指具有特定应用的数据仓库,主要针对某个具有战略意义的应用或者具体部门级的应用,支持用户利用已有的数据获得重要的竞争优势或者找到进入新市场的具体解决方案。

数据集市有两种,即独立的数据集市(Independent Data Mart)和从属的数据集市(Dependent Data Mart)。

### 3. 数据集市与数据仓库的差别

(1) 数据仓库是基于整个企业的数据模型建立的,它面向企业范围内的主题。而数据集市是按照某一特定部门的数据模型建立的,由于每个部门有自己特定的需求,因此,它们对数据集市的期望也不一样。

(2) 部门的主题与企业的主题之间可能存在关联,也可能不存在关联。数据仓库中存储整个企业内非常详细的数据,而数据集市中数据的详细程度要低一些,包含概要和累加数据要多一些。

(3) 数据集市的数据组织一般采用星型模型。大型数据仓库的数据组织,如 NCR 公司采用第三范式。

### 4. 数据集市的特性

数据集市有如下特性:

(1) 规模是小的;
(2) 特定的应用;
(3) 面向部门;
(4) 由业务部门定义,设计和开发;
(5) 由业务部门管理和维护;
(6) 快速实现;
(7) 价格较低廉;
(8) 投资快速回收;
(9) 工具集的紧密集成;

(10) 更详细的、预先存在的数据仓库的摘要子集;

(11) 可升级到完整的数据仓库。

**5. 两种数据集市的结构**

(1) 从属数据集市

从属数据集市的逻辑结构见图 2.2。

所谓从属,是指它的数据直接来自于中央数据仓库。显然,这种结构仍能保持和数据仓库的一致性。一般为那些访问数据仓库十分频繁的关键业务部门建立从属的数据集市,这样可以很好地提高查询的反应速度。

(2) 独立数据集市

独立数据集市的逻辑结构见图 2.3。

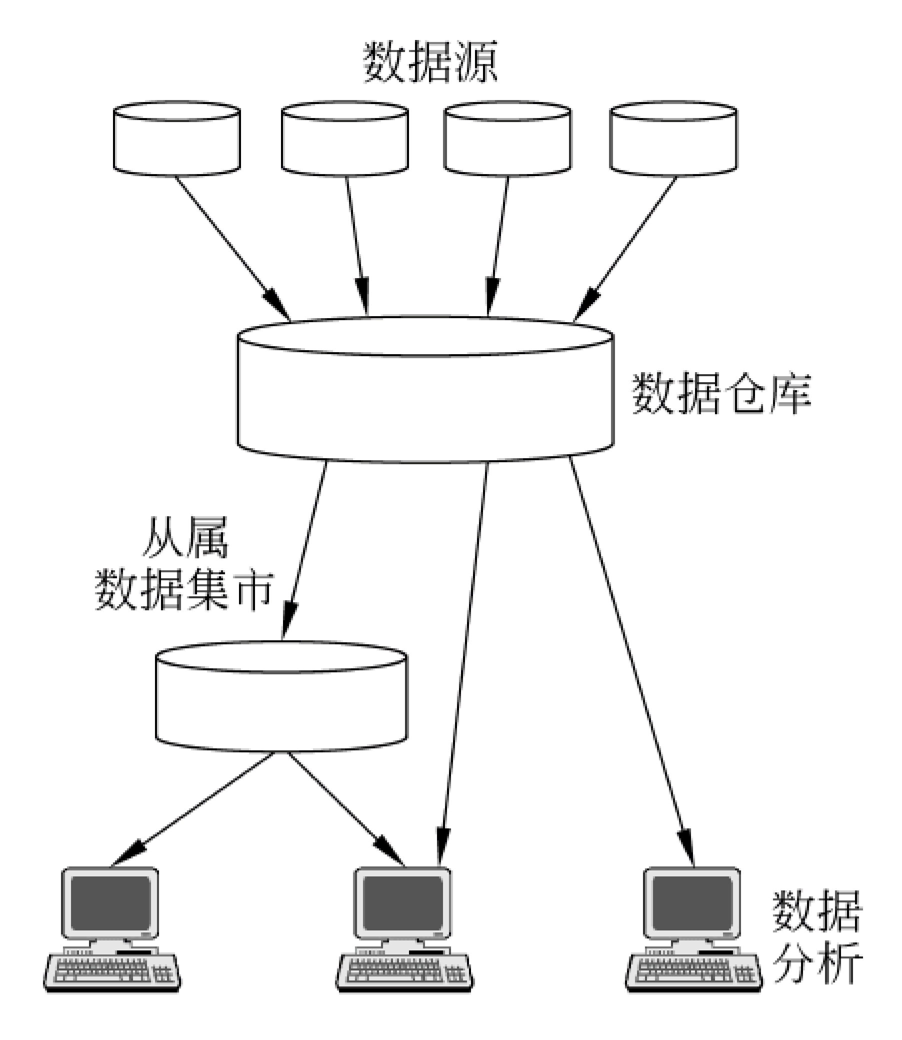



图 2.2　从属数据集市的结构

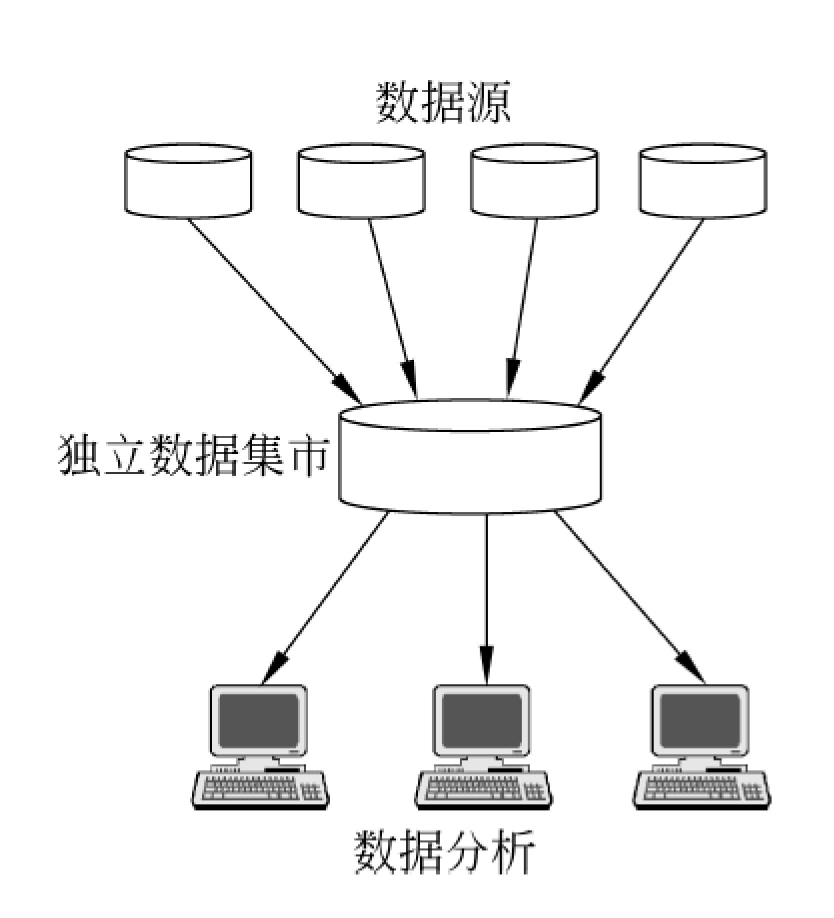



图 2.3　独立数据集市的结构

独立数据集市的数据直接来源于各生产系统。许多企业在计划实施数据仓库时,往往出于投资方面的考虑,最后建成独立数据集市,用来解决个别部门比较迫切的决策问题。从这个意义上讲,它和企业数据仓库除了在数据量大小和服务对象上有所区别外,逻辑结构并无多大区别,这是把数据集市称为部门数据仓库的主要原因。

**6. 关于数据集市的误区**

数据集市是一个数据分支子集,它可以从一个数据仓库中找到,或者是支持一个单独业务单元的决策支持而建立的。甚至企业的大部分战略都可以由数据集市来完成,在这个过程中制定行动方针。但是,在建立一个数据集市之前,企业应该知道几个关于数据集市的不切实际的看法。

(1) 单纯用数据量大小来区分数据集市和数据仓库

用大小来判断一个企业是在实施数据仓库还是数据集市的做法是很片面的。尺寸大小不是数据集市的本质特征,真正的问题在于,数据集市(它可能是一个数据仓库的子集)的数据模型一定是满足应用的特定需求的。

(2) 简单地理解数据集市容易建立

数据集市的确比数据仓库的复杂程度低一些,因为它只针对某一需要解决的特定的商业问题,但是围绕数据获取的很多复杂问题并没有减少。

数据集市要从多个数据源中提取数据,这个过程很耗时,因为这个过程与建立一个数据仓库一样,需要相同的计划和管理,并且需要把数据模型化。

(3) 数据集市很容易升级成数据仓库

事实上,数据集市针对特殊的业务需要,不可能很容易地伸缩。如果没有事先扩展数据模型,追加数据是非常困难的。例如,一个数据集市可以很快找到最畅销款式的鞋的销售数字,为了增加关于这种鞋的信息,比如新顾客的百分比,就需要新的数据模型,这种数据集市的扩充是困难的。

## 2.1.3 数据仓库系统结构

数据仓库系统由数据仓库(DW)、仓库管理和分析工具三部分组成,其结构形式如图 2.4 所示。

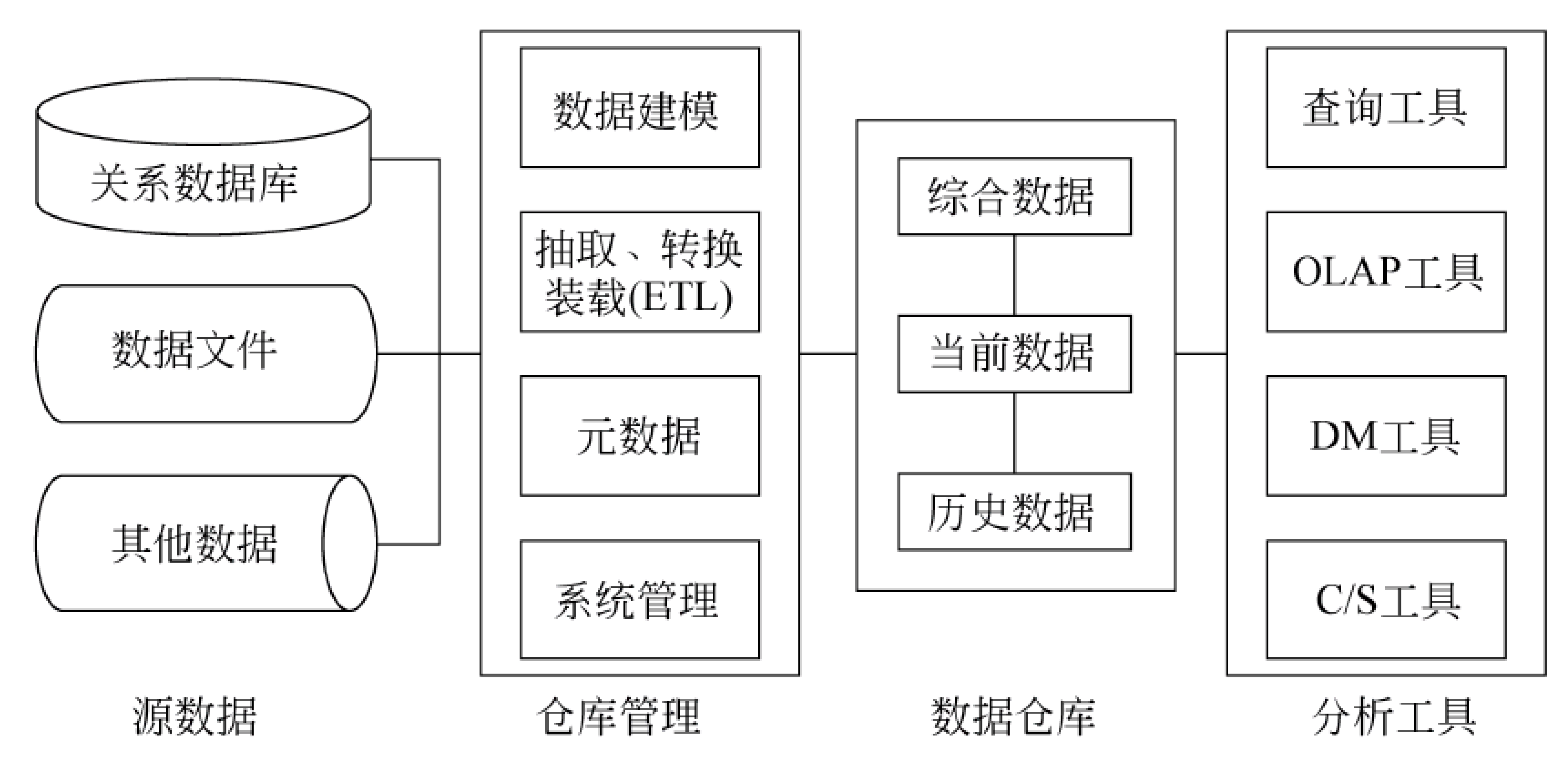



图 2.4 数据仓库系统结构图

数据仓库的数据来源于多个数据源。源数据包括企业内部数据、市场调查报告以及各种文档之类的外部数据。

### 1. 仓库管理

仓库管理包括数据建模;数据抽取、转换、装载(ETL);元数据;系统管理等四部分。

(1) 数据建模

数据建模是建立数据仓库的数据模型(Data Model)。数据模型是现实世界数据特征的抽象。数据模型一般包括数据结构和数据操作。数据结构包括数据类型、内容、数据之间的关系,它是数据模型的静态描述。数据操作是对数据仓库中数据所允许的操作,如检索、计算等。

数据仓库的数据模型,按数据仓库设计过程分为概念数据模型、逻辑数据模型和物理数据模型。

数据仓库的数据模型不同于数据库的数据模型体现在以下方面:

① 数据仓库的数据模型的数据只为决策分析用，不包含那些纯事务处理的数据。

② 数据仓库的数据模型中增加了时间属性的代码数据。

③ 数据仓库的数据模型中增加了一些导出数据，如综合数据等。

数据仓库的数据建模是使建立的物理（存储）数据模型能适应决策用户使用的逻辑数据模型。

(2) 数据抽取、转换、装载(ETL)

数据仓库中的数据，是通过在源数据中抽取数据，按数据仓库的逻辑数据模型的要求进行数据转换，再按物理数据模型的要求装载存储到数据仓库中去的。

数据抽取、转换、装载(ETL)是建立数据仓库的重要步骤，也是一项烦琐、耗时且费劲的工作，需要花费开发数据仓库 70%的工作量。

(3) 元数据

元数据在数据仓库中扮演了一个新的重要角色。元数据不仅是数据仓库的字典，要指导数据的抽取、转换、装载(ETL)工作，还要指导用户使用数据仓库。

(4) 系统管理

系统管理包括数据管理、性能监控、存储器管理以及安全管理等。

数据管理包括为适应竞争变化的业务需求更新数据、清理脏数据、删除休眠数据等工作。

系统对性能的监控是搜集和分析系统性能的信息，确定系统是否达到了所确定的服务水平。

存储器管理是使数据仓库的存储器要适应数据量的增长需求，实现用户的快速检索。

安全管理是保证应用程序的安全以及数据仓库访问的安全。

**2. 分析工具**

由于数据仓库的数据量大，因此必须有一套功能很强的分析工具集来实现从数据仓库中提供辅助决策的信息，完成决策支持系统(DSS)的各种要求。

(1) 查询工具

数据仓库的查询不是指对记录级数据的查询，而是指对分析要求的查询。以图形化方式展示数据，可以帮助了解数据的结构、关系以及动态性。

(2) 多维数据分析工具(OLAP 工具)

通过对多维数据进行快速、一致和交互性的存取，有利于用户对数据进行深入的分析和观察。

多维数据的每一维代表对数据的一个特定的观察视角，如时间、地域、业务等。

(3) 数据挖掘工具(DM 工具)

从大量数据中挖掘具有规律性的知识，需要利用数据挖掘中的各种不同算法。

(4) 客户/服务器(C/S)

数据仓库一般都是以服务器(Server)形式在网络环境下提供服务，能对多个客户(Client)同时提供服务。

### 2.1.4 数据仓库的运行结构

数据仓库应用是一个典型的客户/服务器(C/S)结构形式,如图 2.5 所示。数据仓库采用服务器结构,客户端所做的工作有客户交互、格式化查询、结果显示、报表生成等。服务器端完成各种辅助决策的 SQL 或 MDX(见 3.4.5 节)查询、复杂的计算和各类综合功能等。

现在,越来越普通的一种形式是三层 C/S 结构形式,即在客户端与数据仓库服务器之间增加一个多维数据分析(OLAP)服务器,如图 2.6 所示。




图 2.5 数据仓库的 C/S 结构　　图 2.6 数据仓库应用的三层 C/S 结构

OLAP 服务器将加强和规范化决策支持的服务工作,集中和简化了数据仓库服务器的部分工作,即 OLAP 服务器从数据仓库服务器中抽取数据,在 OLAP 服务器中转换成客户端用户要求的多维视图,并进行多维数据分析,将分析结果传送给客户端。这种结构形式工作效率更高。

## 2.2 数据仓库数据模型

数据仓库不同于数据库。数据仓库的逻辑数据模型是多维结构的数据视图,也称多维数据模型。

在多维数据模型中,主要数据是实际数值,如销售量、投资额、收入等。而这些数值是依赖于一组"维"的,这些维提供了实际值的上下文关系。例如销售量与城市、商品名称、销售时间有关,这些相关的维唯一决定了这个销售实际值。因此,多维数据视图就是在这些维构成的多维空间中存放着数字实际值。图 2.7 中的小格内存储的数据可以假设为商品的销售量。

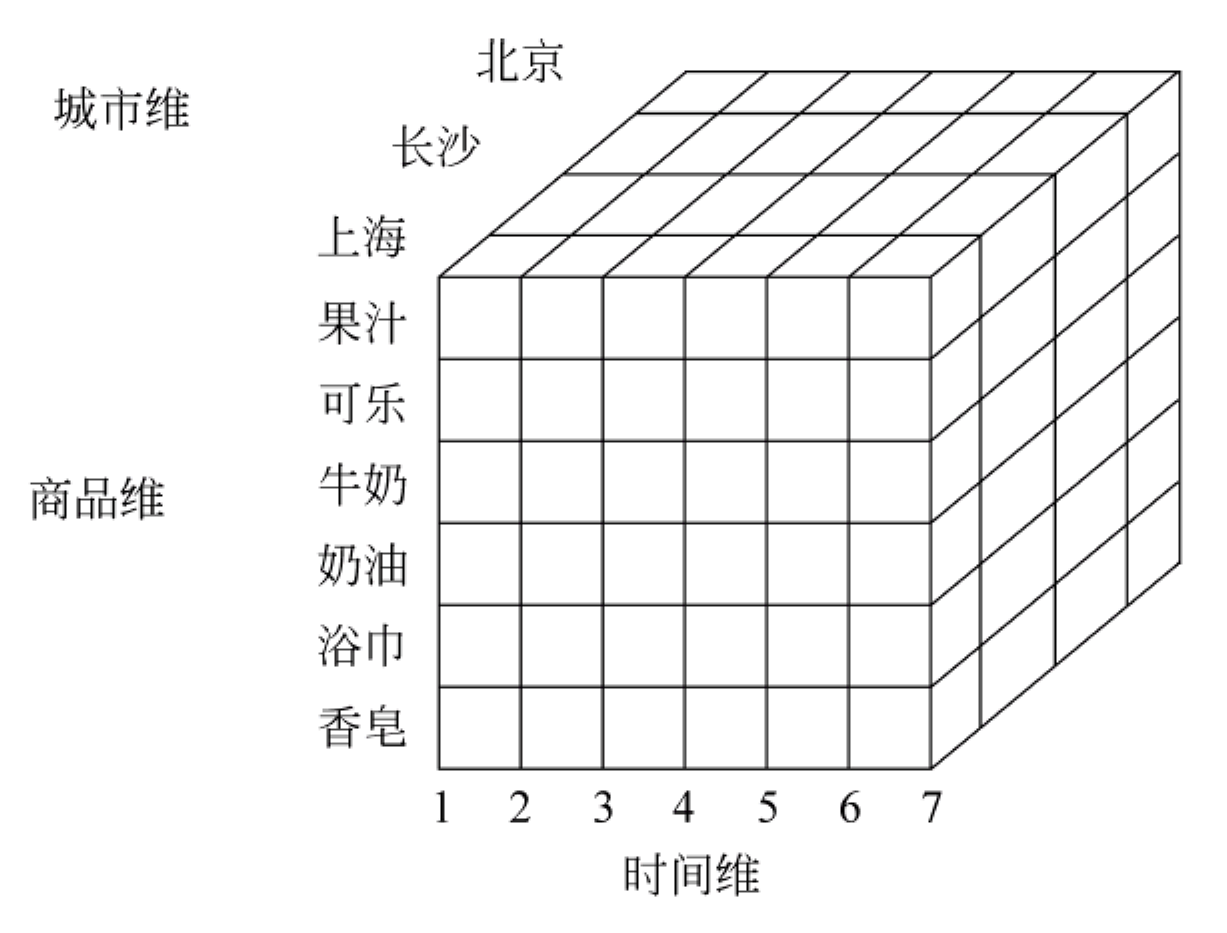



图 2.7 数据仓库的数据模型

多维数据模型的另一个特点是对一个或多个维所完成的集合运算,例如对总销售量按城市进行计算和排序。这些运算还包括对于同样维的实际值进行比较(如销售与预算)。一

般来说,时间维是一个有特殊意义的维,它对决策中的趋势分析很重要。

对于逻辑数据模型,可以使用不同的存储机制和表示模式来实现多维数据模型。目前,使用的多维数据模型主要有星型模型、雪花模型、星网模型、第三范式等。

### 2.2.1 星型模型

大多数的数据仓库都采用“星型模型”。星型模型是由“事实表”(大表)以及多个“维表”(小表)所组成的。“事实表”中存放着大量关于企业的事实数据(数字实际值),对象(元组)个数通常都很大,而且非规范化程度很高。例如,多个时期的数据可能会出现在同一个表中。“维表”中存放描述性数据,维表是围绕事实表建立的较小的表。

一个星型数据模型实例如图 2.8 所示。

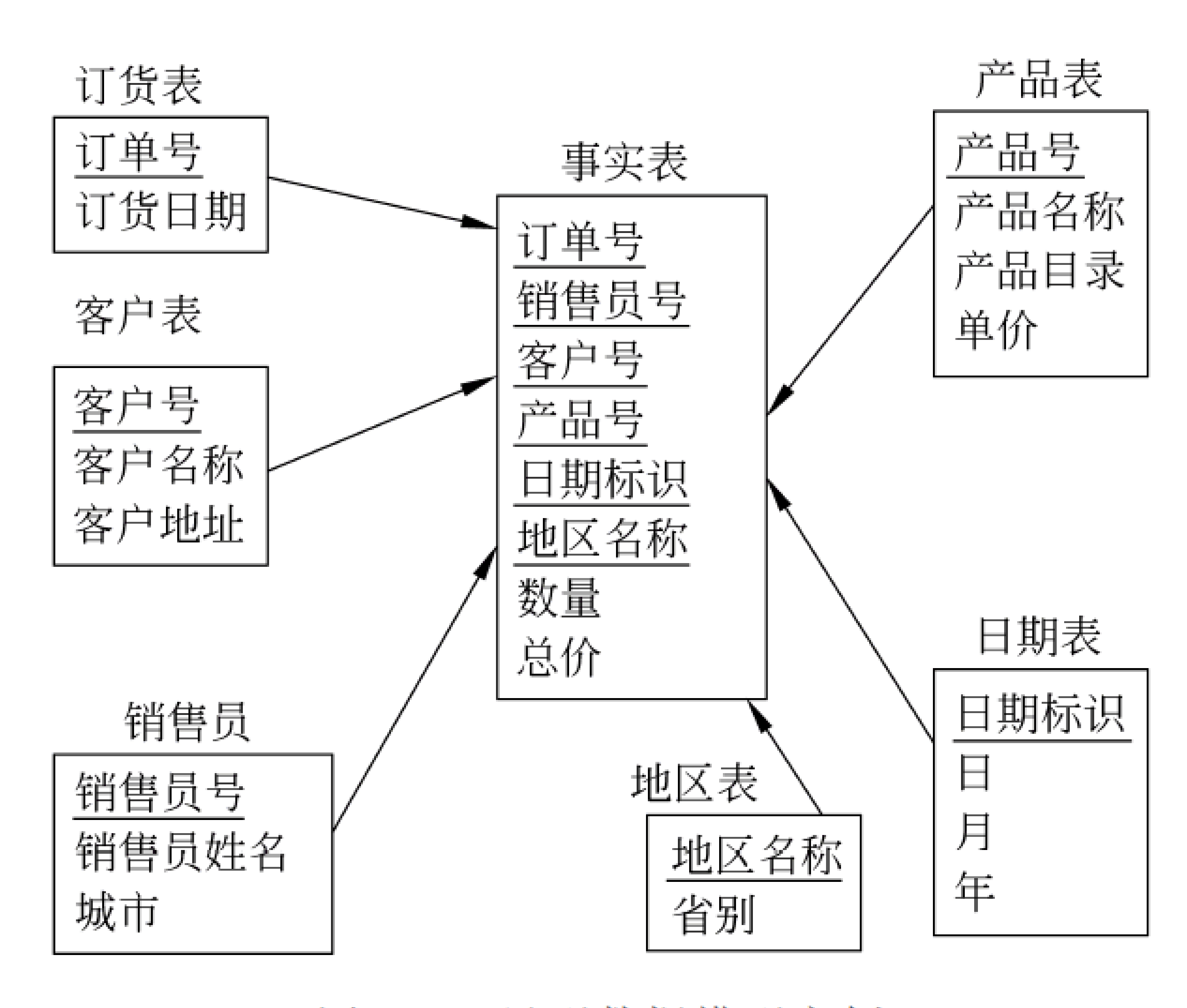



图 2.8 星型数据模型实例

事实表有大量的行(元组),然而维表相对来说有较少的行(元组)。星型模型存储情况如图 2.9 所示。

星型模型存取数据速度快,主要在于针对各个维做了大量的预处理,如按照维进行预先的统计、分类、排序等,再如按照汽车的型号、颜色、代理商进行预先的销售量统计,做报表时速度会很快。

星型结构与规范化的关系数据库设计相比较,存在一些显著的优点:

星型模型是非规范化的,以增加存储空间代价,提高了多维数据的查询速度。而规范化的关系数据库设计是使数据的冗余保持在最少,并减少当数据改变时系统必须执行的动作。

星型模型也有缺点:

当业务问题发生变化,原来的维不能满足要求时,需要增加新的维。由于事实表的主键由所有的维表的主键组成,因此这种维的变化带来数据变化将是非常复杂、非常耗时的。星型模型的数据冗余量很大。

### 2.2.2 雪花模型

雪花模型是对星型模型的扩展,雪花模型对星型模型的维表进一步层次化。原来的各

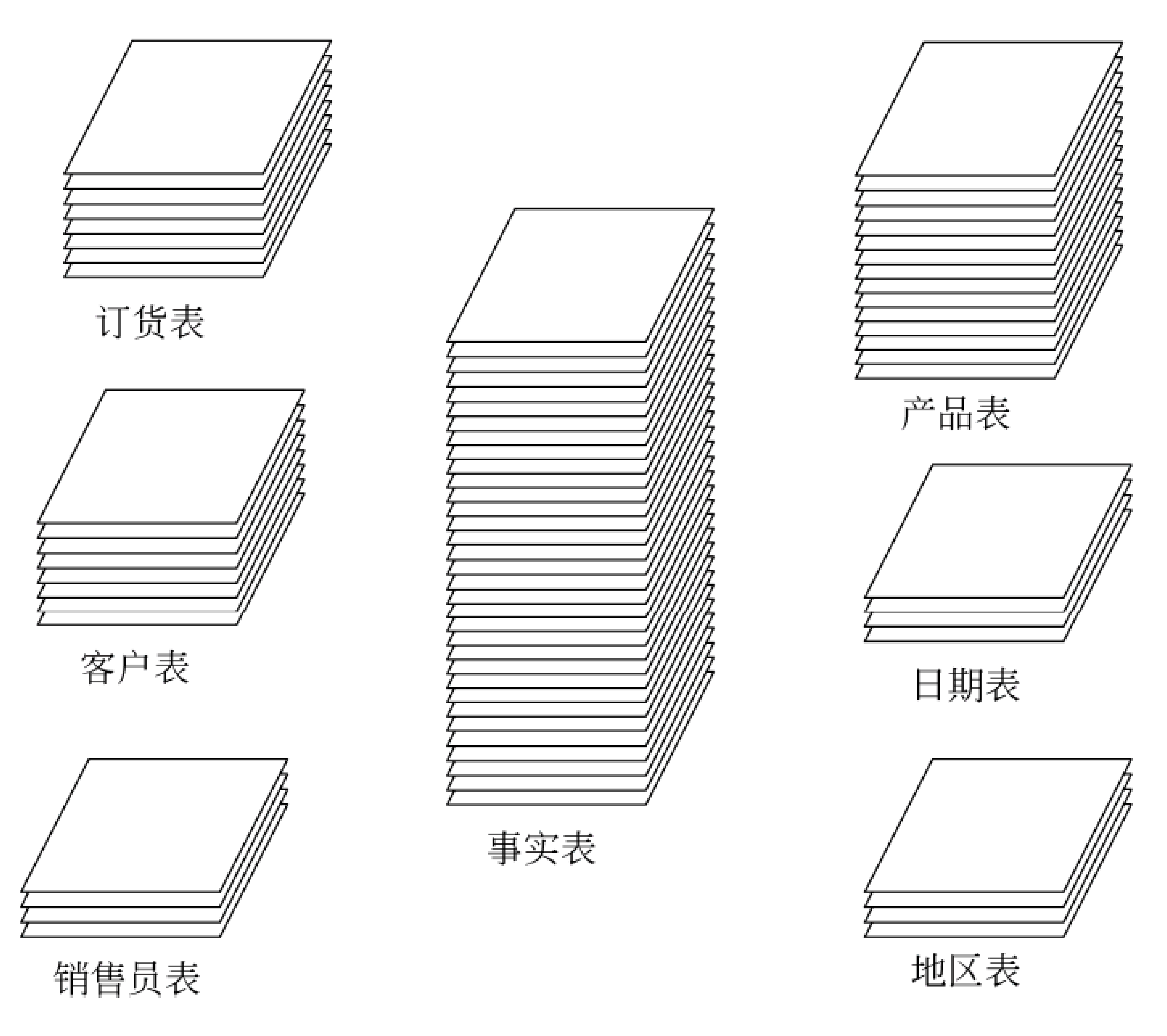



图 2.9　星型模型数据存储情况示意图

维表可能被扩展为小的事实表，形成一些局部的“层次”区域。它的优点是最大限度地减少数据存储量，以及把较小的维表联合在一起来改善查询性能。

雪花模型增加了用户必须处理的表的数量，增加了某些查询的复杂性。但这种方式可以是系统更进一步专业化和实用化，同时降低了系统的通用程度。前端工具将用户的需求转换为雪花模型的物理模式，完成对数据的查询。

在雪花模型中能够定义多重“父类”维来描述某些特殊的维表。比如，在时间维上增加了月维和年维，通过查看与时间有关的父类维，能够定义特殊的时间统计信息，如销售月统计、销售年统计等。

在图 2.8 所示的星型模型的数据中，对“产品表”“日期表”“地区表”进行扩展形成雪花模型数据如图 2.10 所示。使用数据仓库的工具完成一些简单的二维或三维查询，既能够满足用户对复杂的数据仓库查询的需求，又能够完成一些简单查询功能而不用访问过多的数据。

### 2.2.3　星网模型

每个数据仓库都包含了多个星型模型的结构。每一个星型模型都在事实表中保存了一些指标，为特定的目的服务。多个相关的星型模型通过相同的维表连接起来形成网状结构，称为星网模型。在大多数星网模型中，各个事实表共享的维表是时间维。

构造星网模型有几种情况：有的是增加汇总事实表和衍生的维表形成星网模型，有的是构造相关的事实表形成星网模型。

如电话公司需要建立两个事实表，一个事实表跟踪单独的电话事务，它能回答“节假日电话收益与工作日电话收益的对比情况”等类问题；一个事实表累计用户电话支出情况，它能回答“某个用户在某段时间内的电话余额”等类问题。该电话公司星网模型实例如图 2.11 所示。

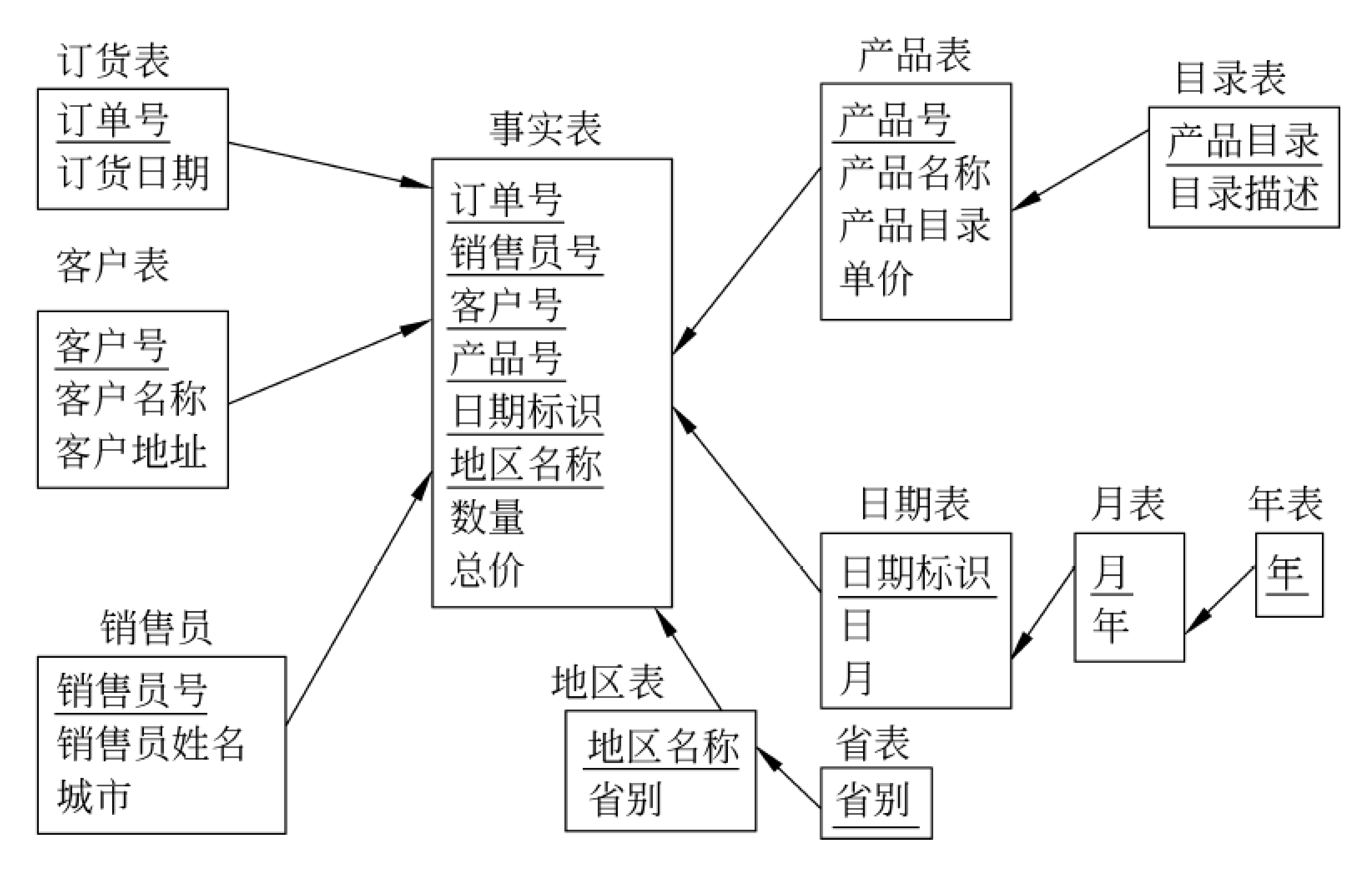



图 2.10　雪花数据模型实例

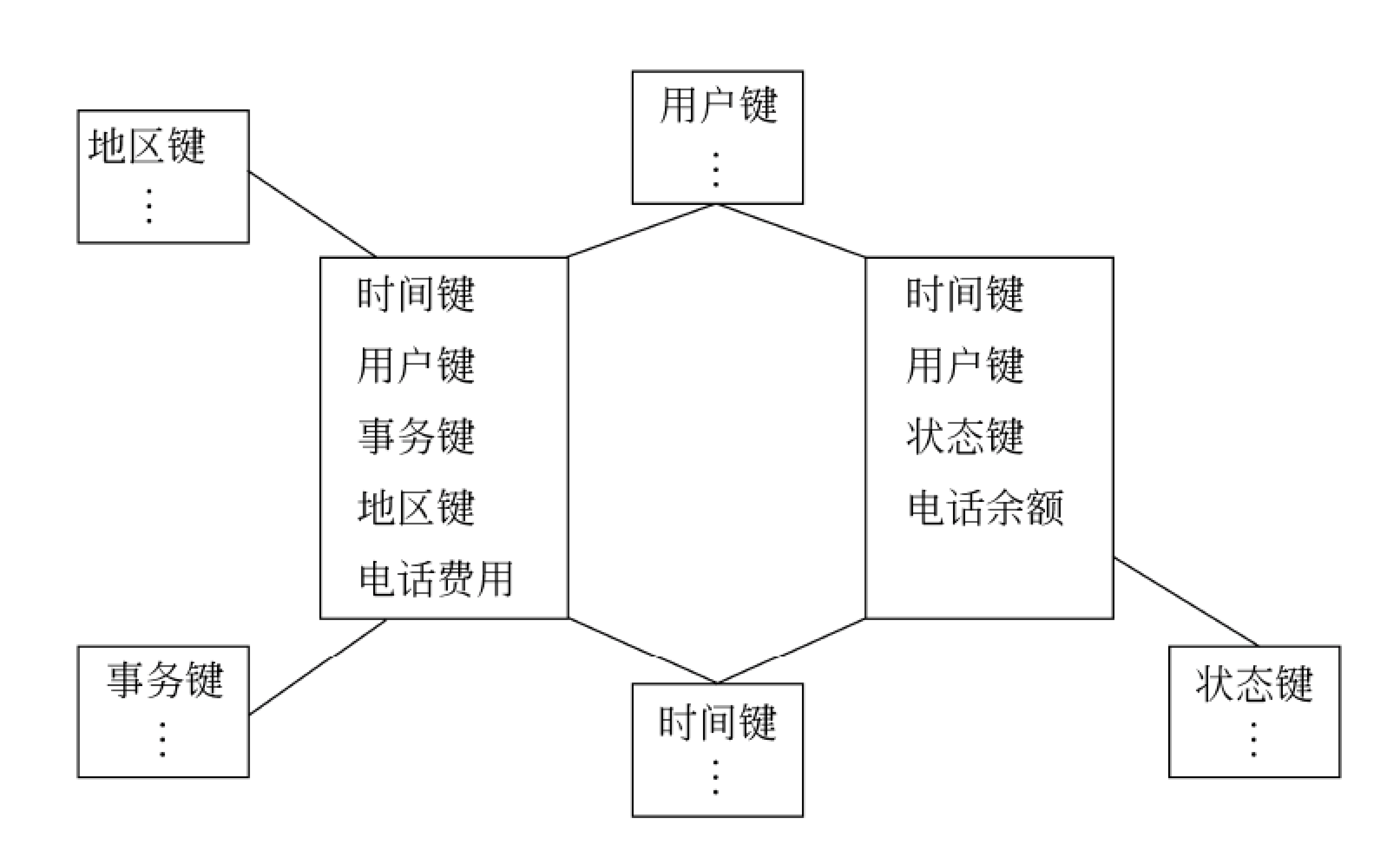



图 2.11　电话公司星网模型实例

## 2.2.4　第三范式

范式实际上是传统的关系数据库的设计理论。一个规范化的关系模式应该准确地反映所描述的数据实体，避免冗余、异常（插入异常、删除异常、更新异常）等问题。

通常按照属性间依赖情况来区分关系规范化的程度，现有第一范式到第五范式。

第三范式（3NF）的作用是解决数据冗余，数据被分割成多个实体，实体在数据库中就用表来表示，使用 3NF 会形成比较复杂的关系表，但它适合于操作型处理，如进行 update 和 insert 等操作。

数据仓库可以按第三范式进行逻辑数据建模。它不同于星型模型之处在于，把事实表和维表的属性作为一个实体都集中在同一数据库表中，或分成多个实体用多个表来表示，每个表按第三范式组织数据。它减少了维表中的键和不必要的属性。

著名的 NCR 数据仓库公司采用了第三范式的逻辑数据模型。现在有很多大型的企业数据仓库系统中都同时采用了第三范式和星型模型，即用第三范式来描述数据仓库系统后

台的详细数据存储关系,在此基础上,再根据特定的分析需求建立适当的星型模型,用于刷新 OLAP 服务器的立方体(Cube),以方便前端数据展现和预定义的多维分析。

星型模型的设计模式适用于决策分析应用。星型模型与第三范式存储的数据信息是一样的,但它更方便用户理解数据,更适合对数据的多维查询操作。

星型模型在进行多维数据分析时,在不超过预定义的维度范围内,速度是很快的。但是,如果超出了预定义的维度,增加维度将是很困难的事情。

第三范式对于海量数据(如 TB 级)且需要处理大量的动态业务分析时,就显示出了它的优势。

## 2.3 数据抽取、转换和装载

数据仓库的数据来源于多个数据源,主要是企业内部数据(用于企业的事务处理,也称操作型数据)、存档的历史数据、企业的外部数据(本行业的统计数据以及竞争者的市场占有率数据等)。这些数据源可能是在不同的硬件平台上,使用不同的操作系统。源数据以不同的格式存放在不同的数据库中。

数据仓库需要将这些源数据经过集成的过程,存储到数据仓库的数据模型中。具体来说,数据仓库的数据获取需要经过抽取(Extraction)、转换(Transform)、装载(Load)三个过程,即 ETL 过程。

经过 ETL 过程,将源系统中的数据改造成有用的信息存储到数据仓库中。例如,ETL 过程将统一各源系统中数据的变量名称,转换和集成所有产品的销售情况数据,装载到数据仓库的销售事实表和相关维表中。在用户查询时,在事实表中提供销售数量与金额的同时,在产品维度表中提供产品目录,在商店维度中提供商店名单,在时间维度中提供日期。这种查询方便了情况对比和决策分析。

ETL 过程在开发数据仓库时,占去 70%的工作量。ETL 过程的主要步骤概括为:

(1) 决定数据仓库中需要的所有的目标数据;

(2) 决定所有的数据源,包括内部和外部的数据源;

(3) 准备从源数据到目标数据的数据映射关系;

(4) 建立全面的数据抽取规则;

(5) 决定数据转换和清洗规则;

(6) 为综合表制定计划;

(7) 组织数据缓冲区域和检测工具;

(8) 为所有的数据装载编写规程;

(9) 维度表的抽取、转换和装载;

(10) 事实表的抽取、转换和装载。

### 2.3.1 数据抽取

数据抽取工作包括以下内容。

### 1. 确认数据源

对数据源的确认不仅仅是对数据源的简单确认，还包括检查和确定数据源是否可以提供数据仓库需要的数据。该项工作包括：

(1) 列出对事实表的每一个数据项和事实；

(2) 列出每一个维度属性；

(3) 对于每个目标数据项，找出源数据项；

(4) 数据仓库中一个数据元素有多个来源，选择最好的来源；

(5) 确认一个目标字段的多个源字段，建立合并规则；

(6) 确认多个目标字段的一个源字段，建立分离规则；

(7) 确定默认值；

(8) 检查缺失值的源数据。

### 2. 数据抽取技术

(1) 进行数据抽取时要考虑的两种情况

① 当前值。源系统中存储的数据都代表了当前时刻的值。当进行商业交易时，这些数据是会发生变化的。

② 周期性的状态。这类数据存储的是每次发生变化时间的状态。例如，对于每一保险索赔，都要经过索赔开始、确认、评估和解决等步骤，都要考虑有时间说明。

在建立数据仓库时，从某一特定时间开始的最初数据必须迁移到数据仓库中，以使数据仓库开始运转，这是初始装载。在初始装载之后，数据仓库必须保持更新，使变化的历史和状态可以在数据仓库中反映出来。

(2) 两类数据的抽取

① 静态数据的抽取。一般在数据仓库的初始装载时抽取的是静态数据，它代表了某个时刻的快照。

② 修正数据的抽取，也称为追加的数据抽取。修正数据的抽取过程包括特定时刻抽取的数据值。它分为立即型数据抽取(实时的数据抽取)和延缓型的数据抽取。

立即型数据抽取的典型方法是，通过读取交易日志，抽取所有相关交易记录。一般利用复制技术从交易日志中捕获交易日志中的变化数据，从日志传输到目标文件中，并检验数据变化的传输情况，确保复制的成功。

延缓型数据抽取的典型方法是，通过读取源记录中包括日期和时间的标记，抽取更新源记录的数据。如果是没有时间标记的旧数据源，就要通过“快照对比技术”，即通过比较源数据的两个快照来抽取变化的数据。

## 2.3.2 数据转换

数据抽取过程中得到的数据是没有经过加工的数据，不能直接应用于数据仓库，必须经过多种处理，将抽取的数据转换成可以存储在数据仓库中的信息。

**1. 数据转换的基本功能**

(1) 选择。从源系统中选择整个记录或者部分记录。

(2) 分离/合并。对源系统中的记录中的数据进行分离操作或者对很多源系统中选择的部分数据进行合并操作。

(3) 转化。对字段的转化包括对源系统进行标准化和使字段对用户来说是可用和可理解的。

(4) 汇总。数据仓库中需要保存很多汇总数据。这需要对最低粒度数据进行汇总。例如,将零售连锁店需要将每一个收款机的每一笔交易的销售数据汇总为每天每个商店关于每种商品的销售数据。

(5) 清晰化。对单个字段数据进行重新分配和简化的过程,使数据仓库更便利使用。

**2. 数据转换类型**

(1) 格式修正。包括数据类型和单个字段长度的变化。例如在源系统中,产品类型通过代码和名称在数值型和文本类型中表示。不同的源系统将会有所不同,对这些数据类型进行标准化,改变成更有意义的文本值。

(2) 字段的解码。对所有晦涩的编码进行解码。将它们变成用户可以理解的值。例如,对性别的解码,在源系统中有的用 1 和 2 表示,有的用 M 和 F 分别表示男性和女性。

(3) 计算值和导出值。在数据仓库中,有时需要与销售和成本一起计算出利润值。导出字段包括平均每天的收支差额和相关比率。

(4) 单个字段的分离。在旧系统中将客户名称、地址存放在大型文本字段中;姓和名存放在一个字段中;城市、地区和邮政编码存放在一个字段中。在数据仓库中却需要将姓名和地址放在不同的字段中,便利不同要求的分析工作。

(5) 信息的合并。例如,一个产品的信息可能从不同的数据源中获得;产品编码和产品名从一个数据源得到,相关包装类型从另一个数据源中得到,成本数据从第三个数据源中得到。信息合并是产品编码、产品名、包装类型和成本的有机组合,是一个新的实体。

(6) 特征集合转化。例如,在源系统中数据采用 EBCDIC 码,而数据仓库数据采用 ASCII 码,这就需要进行代码集合的转化。

(7) 度量单位的转化。使数据具有相同的标准度量单位。不少国家有自己的度量单位,需要在数据仓库中采用标准度量单位。

(8) 日期/时间转化。日期和时间的表示应该转化成国际标准格式。例如 2005 年 10 月 15 日在美国表示成 10/15/2005,而在英国表示为 15/10/2005。标准格式为 15 OCT 2005。

(9) 汇总。这种类型的转换是创建数据仓库的汇总数据。汇总数据适合于客观战略性的查询。

(10) 关键字重新构造。在源系统中关键字可能包含很多项的内容,如产品编码包括仓库代码、销售区域、产品编码等多项内容。在数据仓库中,关键字要发生变化,转换成适合于事实表和维表的普通键值。

### 3. 数据整合和合并

数据仓库的数据是从很多不同的分散的源系统中的源数据集成起来的。各源系统采用不同的命名方式和不同的数据标准。数据整合和合并是将相关的源数据组合成一致的数据结构，装入数据仓库。具体表现为：

(1) 实体识别问题

例如，一个数据仓库的数据来源于三个不同的客户系统：一个是订单登记系统，一个是客户服务支持系统，一个是市场系统。这三个系统中对相同客户可能分别有不同的键码。

在数据仓库中，需要为每一个客户建立一个记录，这就必须从三个源系统中得到同一客户的数据，将它们组合成一条单独的记录。这是客户实体识别问题。

进行数据转换时，需要让用户参与这个过程，帮助对实体的识别，并设计算法，将三个系统中得到的记录进行匹配，建立统一的记录集合。

(2) 多数据源相同属性不同值的问题

例如，假设产品的单位成本可能从两个系统中得到，在特定的时间间隔内对成本值进行计算和刷新，由于两个系统中得到的成本存在一些差别，数据仓库应该从哪个系统中取得成本呢？

有三种方法：

① 分别给这两个系统不同的优先权，取高优先权的成本数据；

② 根据最新的刷新日期来选择其中一个源系统的成本数据；

③ 根据其他相关字段来选择合适的源系统的成本数据。

### 4. 如何实施转换

完成数据转换工作一般采用两种方式：自己编写程序实现数据转换和使用转换工具。

(1) 自己编写程序实现数据转换

在明确了数据转换的类型和数据整合与合并的内容以后，一般具有编程能力的程序员和分析师都可以编写数据转换程序。

这种方式会带来复杂的编程和测试。

(2) 使用转换工具

使用自动的工具会提高效率和准确性。当确定数据转换参数和规则时，将它作为元数据存储在工具中，工具就能按元数据的说明有效地完成数据转换工作。这是使用数据转换工具的主要优点。

## 2.3.3 数据装载

一旦创建了装载映像，数据转换功能就结束了，接下来的是数据装载。它将转换好的数据存储到数据仓库的数据库中去。

数据装载包括数据装载方式和数据装载类型。

#### 1. 数据装载方式

（1）基本装载

按照装载的目标表，将转换过的数据输入到目标表中去。若目标表中已有数据，装载时会先清除这些数据，再装入新数据。目标表可以是事实表或维表。

（2）追加

如果目标表中已经存在数据，追加过程在保存已有数据的基础上增加输入数据。当一个输入数据记录与已经存在的记录重复时，输入记录可能可以作为副本增加进去，或者丢弃新输入数据。

（3）破坏性合并

当输入数据记录的主键与一条已经存在的记录的键互相匹配时，用新输入数据更新目标记录数据。如果输入记录是一条新的记录，没有任何与之匹配的现存记录，那么就将这条输入记录添加到目标表中。

（4）建设性合并

当输入记录主键与已有记录的键相匹配时，保留已有的记录，增加输入的记录，并标记为旧记录的替代。

#### 2. 数据装载类型

数据装载类型包括三种：最初装载、增量装载和完全刷新。

（1）最初装载

这是第一次对整个数据仓库进行装载。在装载工作完成以后，建立索引。

（2）增量装载

由于源系统的变化，数据仓库需要装载变化的数据，这就是增量装载。

在建设性合并的装载方式中，对增加的输入记录中标记了旧记录的替代。这可以作为增量装载的方法。

当已装入的记录数据必须被改正后的数据记录取代时，要采用破坏性合并的装载方式作为增量装载的方法。

（3）完全刷新

这种类型的数据装载用于周期性重写数据仓库。有时，也可能对一些特定的表进行刷新。

完成刷新与初始装载比较相似。不同点在于在完全刷新之前，目标表中已经存在数据。

### 2.3.4 ETL 工具

目前市场上有三类 ETL（数据抽取、转换、装载）工具。

#### 1. 数据转换引擎

这类工具根据用户定义的时间间隔，从一组指定的源系统中抽取数据，执行复杂的数据转换，将结果导入到目标表中。这类工具使用户选择最合适的数据转换方法，实施完全更新和增量装载。

这类工具的功能涵盖了整个 ETL 过程。

### 2. 通过复制捕获数据

这类工具中大部分使用由数据库管理系统维护的交易日志。在交易日志中捕获的源系统的变化,可以近乎实时地在数据准备区域被复制,等待进一步的处理。

### 3. 代码生成器

这类工具根据用户提供的数据源的参数和目标输出以及商业规则,能自动生成数据抽取和转换程序,完成 ETL 过程。

这类工具的自动化程度较高。

对数据仓库的数据抽取、数据转换和数据装载过程,选择 ETL 工具时,需要考虑以下特征:

(1) 从多种关系型数据库中抽取数据;
(2) 从旧数据库、索引文件和平面文件中抽取数据;
(3) 源字段和目标字段从一种格式向另一种格式进行数据转换;
(4) 执行标准转化、重定义键和结构性变化;
(5) 提供从数据源到目标的检查轨迹;
(6) 抽取和转换中商业规则的应用;
(7) 将源系统中的几个记录组合成一个整合的目标记录;
(8) 元数据的记录和管理。

# 2.4 元 数 据

## 2.4.1 元数据的重要性

元数据在数据仓库的建造、运行中有着极其重要的作用。元数据描述了数据仓库的数据和环境,遍及数据仓库的所有方面,是整个数据仓库的核心。

元数据可分为四类,分别为关于数据源的元数据、关于数据模型的元数据、关于数据仓库映射的元数据和关于数据仓库使用的元数据。

下面是元数据的一个例子,它定义了数据仓库中的一个表,如表 2.1 所示。

**表 2.1 元数据例**

| Table | 逻 辑 名 | 顾 客 |
|---|---|---|
| | 定义 | 购买商品的个人或组织 |
| | 物理存储 | DB. table(数据库表) |
| | 建立日期 | 2008 年 1 月 15 日 |
| | 最后更新日期 | 2010 年 1 月 20 日 |
| | 更新周期 | 每月 |
| | 表编辑程序名 | ABC(程序名) |

最基本的元数据相当于数据库系统中的数据字典。由于数据仓库与数据库有很大的不同，因此元数据的作用远不是数据字典所能相比的。元数据在数据仓库中有着举足轻重的作用，它不仅定义了数据仓库有什么，指明了数据仓库中数据的内容和位置，刻画了数据的抽取和转换规则，存储了与数据仓库主题有关的各种商业信息，而且整个数据仓库的运行都是基于元数据的，如数据的修改、跟踪、抽取、装入、综合等。

有两类人会用到元数据：最终用户（包括商业分析员）和 IT 人员（包括开发人员和管理人员）。

**1. 最终用户**

数据仓库的用户希望从数据仓库获取信息来回答以下问题：

- 每个商店各种产品每天的销售数量和金额是按照每一笔交易，还是按照汇总数据存储？
- 销售情况能够按照产品、促销、商店和月份进行分析吗？
- 当月的销售能与去年同期销售对比吗？
- 销售情况能与预期目标进行比较吗？
- 利润率是如何计算的？商业规则有哪些？
- 销售区域是如何划定的？需要分析的两个区域包含了哪些地区？
- 销售情况的数据从何而来？来自哪些源系统？
- 销售数据是什么时候的？这些数据多久更新一次？

最终用户需要的元数据有数据内容、汇总数据、商业维度、商业指标、浏览路径、源系统、外部数据、数据转换规则、最后更新日期、数据装载和更新周期、查询模板、报表格式、预定义查询和报表、OLAP 数据等。

最终用户需要的元数据也称为商业元数据，它像一幅公路地图，显示了信息所在的地方，以及如何到达那个地方。最终用户通过商业元数据的引导，能够有效地从数据仓库中获得所需要的信息，提高分析效果。

**2. IT 人员**

元数据对数据仓库的开发者和管理者来说都很重要。从开始的数据抽取、数据转换、数据集成、数据清洗、数据准备、数据存储，到查询及报表设计、OLAP 设计以及运行时的管理工作，IT 人员必须能够得到合适的元数据。

IT 人员需要的元数据有：源数据结构、源平台、数据抽取方法、外部数据、数据转换规则、数据清洗规则、准备区域结构、维度模型、初始装载、增量装载、数据汇总、OLAP 系统、Web 访问、查询和报表设计。

IT 人员需要的元数据也称为技术元数据，为负责开发、管理和维护数据仓库服务。技术元数据对 IT 人员来说，就像一个支持技术工作的指南。

### 2.4.2 关于数据源的元数据

它是现有的业务系统的数据源的描述信息。这类元数据是对不同平台上的数据源的物

理结构和含义的描述。具体为：

（1）数据源中所有物理数据结构，包括所有的数据项及数据类型。

（2）所有数据项的业务定义。

（3）每个数据项更新的频率，以及由谁或哪个过程更新的说明。

（4）每个数据项的有效值。

（5）其他系统中具有相同业务含义的数据项的清单。

### 2.4.3 关于数据模型的元数据

这组元数据描述了数据仓库中有什么数据以及数据之间的关系，它们是管理和使用数据仓库的基础。这种的元数据可以支持用户从数据仓库中获取数据。用户可以提出需要哪些表，系统从中选一个表，并得到表之间的关系。通过关系新表，重复该过程。用户能够得到希望的数据。

描述数据仓库中的数据及数据之间的各种复杂关系，元数据要定义以下内容：

（1）I/O对象：支持数据仓库I/O操作的各种对象。元数据要描述该I/O对象的定义、类型、状态、存档（刷新）周期。

（2）关系：两个I/O对象之间关联。这种关联有三种类型：一对一、一对多和多对多。

（3）关系成员：描述每个关系中两个I/O对象的具体角色（在一对多中是父亲还是儿子）、关系度（一对一还是一对多）及约束条件（必须满足还是可选关系）。

（4）关系关键字：描述两个I/O对象的是如何建立关联的。每个关系都是通过I/O对象的关键字来建立的，元数据要指明建立每个关系的相应对象的关键字。

这组元数据定义的数据之间的关系可以用图2.12来表示。

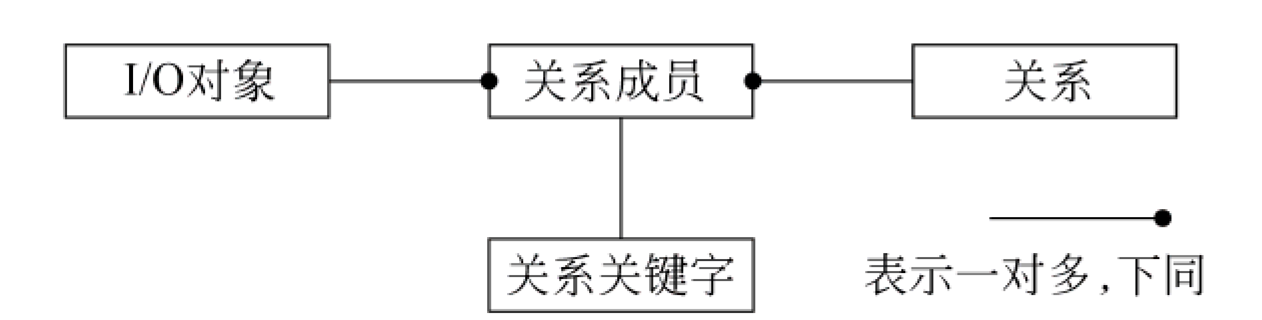



图2.12 数据模型的元数据内容

例如，雇员与技能之间的关系如图2.13所示。

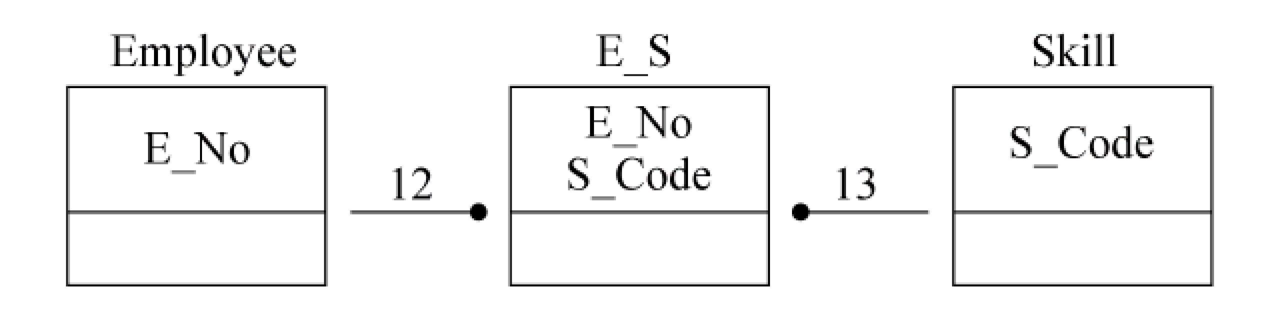



图2.13 雇员与技能之间的关系图

在数据仓库中元数据描述该关系如图2.14所示。

### 2.4.4 关于数据仓库映射的元数据

这类元数据是数据源与数据仓库数据之间的映射。

当数据源中的一个数据项与数据仓库建立了映射关系时，就应该记下这些数据项发生

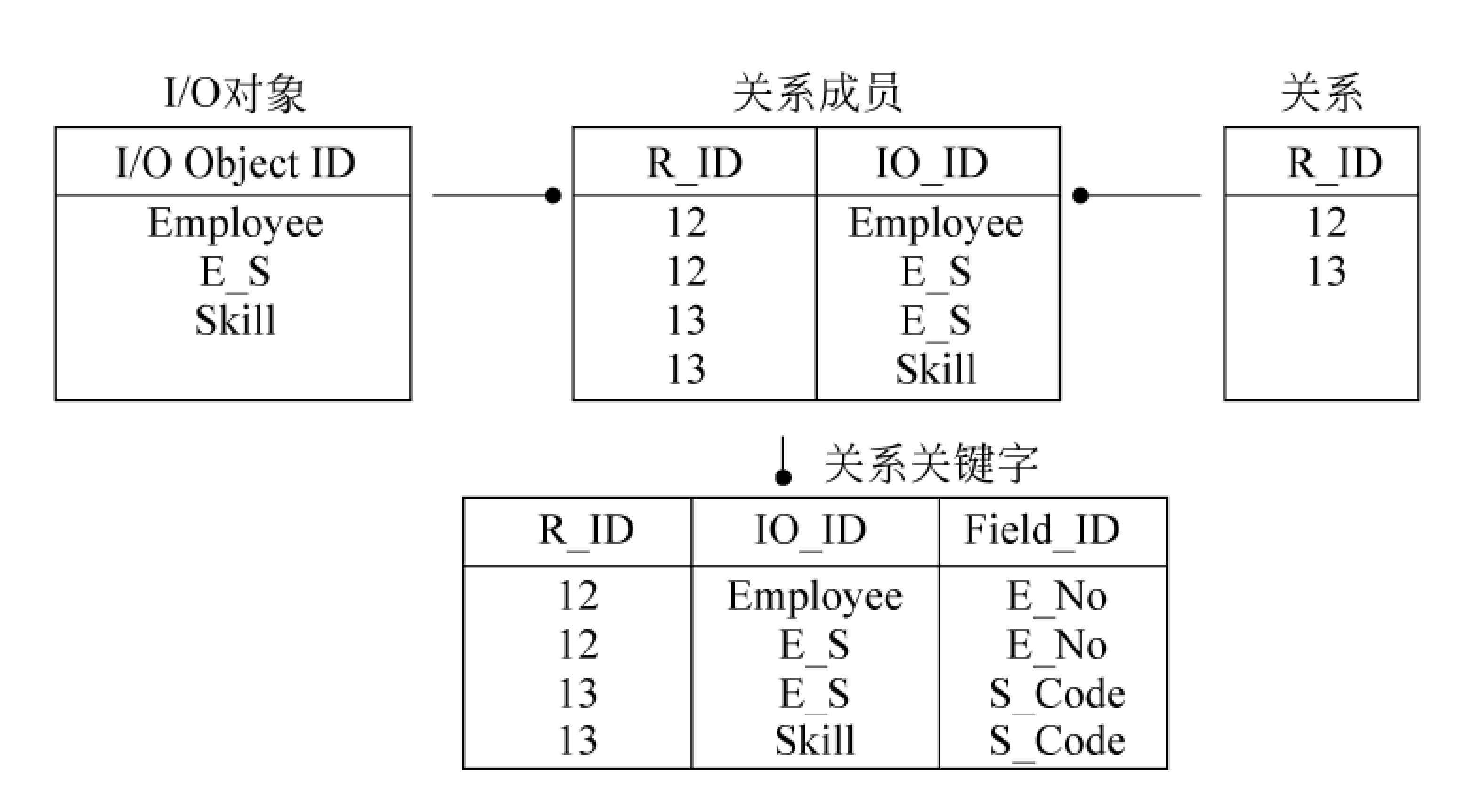



图 2.14　雇员与技能关系的元数据内容

的任何变换或变动，即用元数据反映数据仓库中的数据项是从哪个特定的数据源抽取的，经过了哪些转换、变换和装载过程。

从源系统的数据到数据仓库中的目标数据的转移是一项复杂的工作，其工作量占整个数据仓库开发的 70%。这里主要涉及两个问题。

**1. 抽取工作之间的复杂关系**

一个数据的抽取要经过许多步骤，如图 2.15 所示。

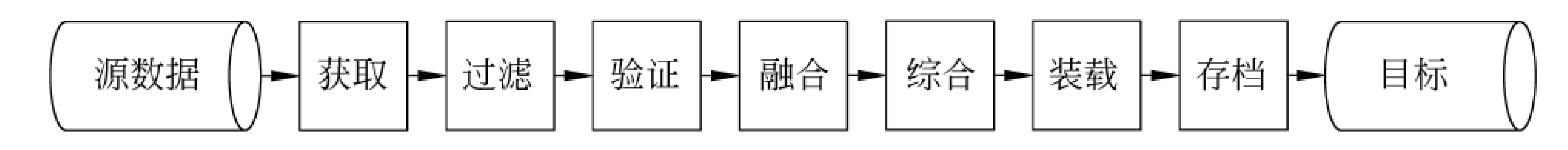



图 2.15　数据抽取工作的步骤

(1) 获取：从外部或内部源数据系统获取对决策支持系统用户有用的数据。
(2) 过滤：过滤掉不需要的内容(如上次抽取后一直没改变的数据)。
(3) 验证：从用户的角度验证数据的质量。
(4) 融合：把本次抽取的数据与数据仓库中的数据进行融合。
(5) 综合：对数据进行综合，生成综合级数据。
(6) 装载：把新数据装入到数据仓库中。
(7) 存档：把新装入的数据单独存为一个文件，以便减少更新操作的数据量。

**2. 源数据与目标数据之间的映射**

源数据与目标数据之间是一种复杂的多对多关系。

元数据要能够描述这些限制所带来的一系列问题。这组元数据要定义的内容如下所示。

(1) 抽取工作：描述每一个抽取工作，并为每一个抽取工作标识其源系统，明确其刷新周期(两次抽取之间的间隔)。

(2) 抽取工作步骤：定义抽取工作中的步骤包括说明每一步的类型(如过滤、验证等)。

(3) 抽取表映射：为每一个抽取步骤建立输入文件/表与输出文件/表之间的关联。

(4) 抽取属性映射：为每一个抽取步骤建立输入表(文件)的属性与输出表(文件)的属性之间的关联。

(5) 记录筛选规则：在抽取工作的每一步骤中进行记录的筛选。如例子：

```
IF Record.Last_Update_Date> '2009_11_01'OR Record.Create_Date> '2009_11_01'
THEN  Reserve(保留)  ELSE  Delete(删除)
```

这类元数据要定义的数据之间的关系可表示如图 2.16 所示。

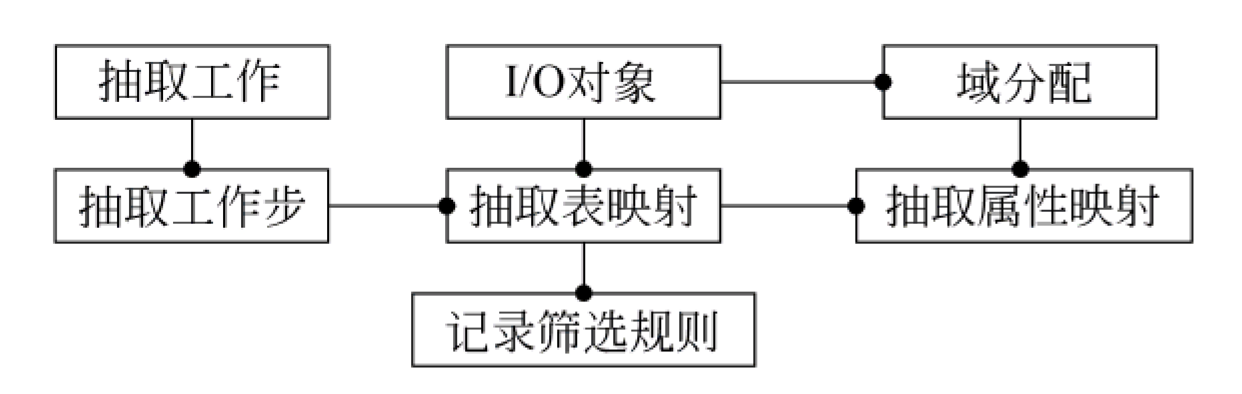



图 2.16 数据映射的元数据内容

这类元数据可以用来生成源代码,以完成数据的转换工作,即完成由操作型数据转换成面向主题的数据仓库的数据。元数据中的抽取表映射和抽取属性映射定义了进行实际抽取转换工作的过程。数据仓库管理核心利用该类元数据所定义的抽取过程生成某种语言的源代码(如 VC),然后编译成可执行的程序以完成数据的抽取工作。

### 2.4.5 关于数据仓库使用的元数据

这类元数据是对数据仓库中信息使用情况的描述。

数据仓库的用户最关心的是两类元数据：

(1) 元数据告诉数据仓库中有什么数据,它们从哪里来,即如何按主题查看数据仓库的内容。

(2) 元数据提供已有的可重复利用的查询语言信息。如果某个查询能够满足他们的需求,或者与他们的愿望相似,他们就可以再次使用这些查询而不必从头开始编程。

更高级的形式是用户通过选择他们要提出的业务问题类型来访问现有的查询,得到相似查询的元数据。

关于数据仓库使用的元数据能帮助用户到数据仓库查询所需要的信息,用于解决企业问题。

## 习 题 2

1. 画出数据仓库结构图,说明各部分内容。

2. 说明数据仓库结构图中包含轻度综合层与高度综合数据层的作用。这些数据为什么不是临时计算出来的?

3. 说明数据集市与数据仓库的区别和联系。

4. 说明数据集市的特点。

5. 画出数据集市的两种结构图,说明它们的不同点。

6．画出数据仓库系统结构图，说明把仓库管理和分析工具作为数据仓库系统的两个独立组成部分的原因。

7．说明仓库管理包含的具体内容。

8．说明分析工具包含的具体内容。

9．画出数据仓库的运行结构图，说明三层 C/S 结构与两层 C/S 结构的不同点。

10．数据仓库的逻辑数据模型有哪些？

11．数据模型与数学模型有什么区别？（提示：数学模型是指运筹学中研究的模型。）

12．说明星型模型有什么好处。

13．说明数据仓库的数据模型为什么含时间维数据。

14．说明雪花模型与星网模型的不同点。

15．第三范式数据模型与星型模型有什么不同？

16．说明第三范式与星型模型的优缺点。

17．简单说明 ETL 过程的主要步骤。

18．说明数据抽取工作的内容。

19．说明数据转换的基本功能。

20．数据转换有哪些类型？

21．数据装载方式与类型有哪些？

22．说明数据库中的元数据以及数据仓库中元数据的不同。

23．什么是关于数据源的元数据？

24．什么是关于数据模型的元数据？

25．什么是关于数据仓库映射的元数据？

26．什么是关于数据仓库使用的元数据？

27．数据仓库中的元数据是如何发挥作用的？

# 第3章　联机分析处理

在数据仓库系统中，联机分析处理（OLAP）是重要的数据分析工具。OLAP 的基本思想是企业的决策者应能灵活地，从多方面和多角度以多维的形式来观察企业的状态和了解企业的变化。

## 3.1　OLAP 概念

在信息爆炸的时代，信息过量几乎成为人人都需要面对的问题。如何才能不被信息的汪洋大海所淹没，从中及时发现有用的知识或者规律，提高信息利用率呢？要想使数据真正成为一个决策资源，只有充分利用它为一个组织的业务决策和战略发展服务才行，否则大量的数据可能会成为包袱，甚至成为垃圾。OLAP 是解决这类问题最有力的工具之一。

OLAP 专门设计用于支持复杂的分析操作，侧重对分析人员和高层管理人员的决策支持，可以应分析人员的要求，快速、灵活地进行大数据量的复杂查询处理，并且以一种直观易懂的形式将查询结果提供给决策制定者，以便他们准确掌握企业（公司）的经营状况，了解市场需求，制定正确方案，增加效益。OLAP 软件以它先进的分析功能和以多维形式提供数据的能力，正作为一种支持企业关键商业决策的解决方案而迅速崛起。

### 3.1.1　OLAP 的定义

在决策活动中，决策人员需要的数据往往不是单一指标的单一的值，他们希望能够从多个角度观察某个指标或者某个值，或者找出这些指标之间的关系。比如，决策者可能想知道“东北地区和西南地区今年一季度和去年一季度在销售总额上的对比情况，并且销售额按10 万～50 万、50 万～100 万，以及 100 万以上分组”。上面的问题是比较有代表性的，决策所需数据总是与一些统计指标如销售总额、观察角度（如销售区域、时间）和不同级别的统计有关，可以将这些观察数据的角度称之为维。可以说决策数据是多维数据，多维数据分析是决策分析的主要内容。但传统的关系数据库系统及其查询工具对于管理和应用这样复杂的数据显得力不从心。

OLAP 是在 OLTP 的基础上发展起来的，OLTP 是以数据库为基础的，面对的是操作人员和低层管理人员，对基本数据的查询和增、删、改等进行处理。而 OLAP 是以数据仓库为基础的数据分析处理。它有两个特点：一是在线性（On Line），体现为对用户请求的快速响应和交互式操作，它的实现是由客户机/服务器这种体系结构在网络环境上完成的；二是多维分析（Multi-dimension Analysis），这也是 OLAP 的核心所在。

OLAP 超越了一般查询和报表的功能，它是建立在一般事务操作之上的另外一种逻辑步骤，因此，它的决策支持能力更强。在多维数据环境中，OLAP 为终端用户提供了复杂的数据分析功能。通过 OLAP，高层管理人员能够通过浏览、分析数据去发现数据的变化趋

势、特征以及一些潜在的信息,从而更好地帮助他们了解商业活动的变化。目前,普遍为人们所接受的 OLAP 的定义有两种。

**1. OLAP 理事会给出的定义**

联机分析处理(OLAP)是一种软件技术,它使分析人员能够迅速、一致、交互地从各个方面观察信息,以达到深入理解数据的目的。这些信息是从原始数据转换过来的,按照用户的理解,它反映了企业真实的方方面面。

企业的用户对企业的观察自然是多维的。例如销售,不仅可从生产这方面看,还与地点、时间等有关,这就是为什么要求 OLAP 模型是多维的原因。这种多维用户视图通过一种更为直观的分析模型进行设计和分析。

OLAP 的大部分策略都是将关系型的或普通的数据进行多维数据存储,以便于进行分析,从而达到联机分析处理的目的。这种多维数据库也被看做超立方体,沿着多个维度存储数据,为用户沿事物的任意的多个维度方便地分析数据。

**2. OLAP 简单定义**

近来,随着人们对 OLAP 理解的不断深入,有些学者提出了更为简要的定义,即联机分析处理是共享多维信息的快速分析(Fast Analysis of Shared Multidimensional Information),它体现了四个特征:

(1) 快速性(fast):用户对 OLAP 的快速反应能力有很高的要求。系统应能在 5s 内对用户的大部分分析要求做出反应,如果终端用户在 30s 内没有得到系统的响应,则会变得不耐烦,改变分析主线索,影响分析的质量。

(2) 可分析性(analysis):OLAP 系统应能处理与应用有关的任何逻辑分析和统计分析。尽管系统需要一些事先的编程,但并不意味着系统事先已将所有的应用都定义好了。

(3) 多维性(multidimensional):多维性是 OLAP 的特点。系统必须提供对数据分析的多维视图和分析,包括对层次维和多重层次维的完全支持。

(4) 信息性(information):不论数据量有多大,也不管数据存储在何处,OLAP 系统都应能及时获得信息,并且管理大容量的信息。

用于实现 OLAP 的技术主要包括网络环境上客户机/服务器体系结构、时间序列分析、面向对象、并行处理、数据存储优化等。

### 3.1.2 OLAP 准则

1985 年以来,关系数据库需求始终受到 E. F. Codd 提出的 12 条规则的影响。1993 年,E. F. Codd 在 *Providing OLAP to User Analysts* 一书中又提出了有关 OLAP 的十二条准则,用来评价分析处理工具,这也是他继关系数据库和分布式数据库提出的两个"十二条准则"后提出的第三个"十二条准则"。由于这些规则最初是对客户研究的结果,所以业界对这个十二条准则褒贬不一。但其主要方面,如多维数据分析、客户/服务器结构、多用户支持及一致的报表性能等方面还是得到了大多数人的认可。E. F. Codd 在文中系统阐述了有关 OLAP 产品及其所依赖的数据分析模型的一系列概念及衡量标准,这对 OLAP 产品的辨别

及后来的发展方向的确立都产生了重要的作用。如今,这十二条规则也成为大家定义 OLAP 的主要依据,被认为是 OLAP 产品应该具备的特征。如今 OLAP 的概念已经在商业数据库领域得以广泛使用,Codd 提出的 OLAP 准则如下。

**1. 多维概念视图**

从用户分析员的角度来看,用户通常从多维角度来看待企业,企业决策分析的目的不同,决定了分析和衡量企业的数据总是从不同的角度来进行的,所以企业数据空间本身就是多维的。因此 OLAP 的概念模型也应是多维的。用户可以简单、直接地操作这些多维数据模型。例如,用户可以对多维数据模型进行切片、切块、改变坐标或旋转模式中的联合(概括和聚集)数据路径。

**2. 透明性**

透明性原则包括两层含义:首先,OLAP 在体系结构中的位置对用户是透明的。OLAP 应处于一个真正的开放系统结构中,它可使分析工具嵌入用户所需的任何位置,而不会对分析工具的使用产生副作用,同时必须保证 OLAP 工具的嵌入不会引入和增加任何复杂性。其次,OLAP 的数据源对用户也是透明的。用户只需使用熟悉的查询工具进行查询,而不必关心 OLAP 工具获取的数据是来自于同质还是异质的数据源。

**3. 可访问性**

OLAP 系统不仅能进行开放的存取,而且还提供高效的存取策略。OLAP 用户分析员不仅能在公共概念视图的基础上对关系数据库中的数据进行分析,而且还可以在公共分析模型的基础上对关系数据库、数据仓库的数据进行分析。要实现这些功能,就要求 OLAP 能将自己的概念视图映射到异质的数据存储上,并可访问数据,还能进行所需的转换以便给出单一的、连贯的、一致的用户视图。另外必须说明的一点就是,物理数据来源于何种系统,这对用户来说应是透明的,进行处理的是 OLAP 工具而不是用户分析员。这是提供 OLAP 工具透明性准则的基础之一。

OLAP 系统应该提供高效的存储策略,使系统只存取与指定分析有关的数据,避免多余的数据存取。

**4. 一致稳定的报表性能**

报表操作不应随维数增加而削弱,即当数据维数和数据的综合层次增加时,提供给最终分析员的报表能力和响应速度不应该有明显的降低,这对维护 OLAP 产品的简易性至关重要。即便是企业模型改变,关键数据的计算方法也无需更改。也就是说,OLAP 系统的数据模型对企业模型应该具有“鲁棒”性。只有做到这一点,OLAP 工具提供的数据报表和所做的预测分析的结果才是可信的。

**5. 客户/服务器体系结构**

OLAP 是建立在客户/服务器体系结构之上的。这要求它的多维数据库服务器能够被

不同的应用和工具所访问,服务器端以最小的代价完成同多种服务器之间的挂接任务,智能化服务器必须具有在不同的逻辑的和物理的数据库间映射并组合数据的能力,还应构造通用的、概念的、逻辑的和物理的模式,从而保证透明性和建立统一的公共概念模式、逻辑模式和物理模式。客户端负责应用逻辑及用户界面。

### 6. 维的等同性

每一数据维在其结构和操作功能上必须等价。可能存在适用于所有维的逻辑结构,提供给某一维的任何功能也应提供给其他维,即系统可以将附加的操作能力授给所选维,但必须保证该操作能力可以授给任意的其他维,即要求维上的操作是公共的。该准则实际上是对维的基本结构和维上的操作的要求。

### 7. 动态的稀疏矩阵处理

OLAP 服务器的物理结构应完全适用于特定的分析模式,创建和加载此种模式是为了提供优化的稀疏矩阵处理。当存在稀疏矩阵时,OLAP 服务器应能推知数据是如何分布的,以及怎样存储才更有效。

该准则包括两层含义:第一,对任意给定的稀疏矩阵,存在一个最优的物理视图,该视图能提供最大的内存效率和矩阵处理能力,稀疏度是数据分布的一个特征,不能适应数据集合的数据分布,将会导致快速、高效操作的失败。第二,OLAP 工具的基本物理数据单元可配置给可能出现的维的子集。同时,还要提供动态可变的访问方法并包含多种存取机制,例如:直接计算地址、B 树索引、导出算法、哈希算法或这些技术的最佳组合。访问速度不会因数据维的多少、数据集的大小而变化。

如果分析要求较为单一和固定,那么确实有可能针对它建立起一个最优的、静态的、具有固定维数的物理模式。但实际上,分析需求的特点就是具有不确定性,所以建立静态模式是不现实的,因此 OLAP 工具必须使得模型的物理模式充分适应指定的维数,尤其是特定模型的数据分布。

### 8. 多用户支持能力

当多个用户在同一分析模式上并行工作,或是在同一企业数据上建立不同的分析模型时,OLAP 工具应提供并发访问、数据完整性及安全性等功能。

实际上,OLAP 工具必须支持多用户也是为了适合数据分析工作的特点。应该鼓励以工作组的形式来使用 OLAP 工具,这样多个用户就可以交换各自的想法和分析结果。

### 9. 非限定的跨维操作

在多维数据分析中,所有维的生成和处理都是平等的。OLAP 工具应能处理维间相关计算。如果计算时需要按语言定义各种规则,此种语言应允许计算和数据操作跨越任意数目的数据维,而不必限制数据单元间的任何关系,也不必考虑每一单元包含的通用数据属性数目。

**10. 直观的数据操作**

OLAP 操作要求直观易懂。如果要重定向联系路径，或在维或行间进行细剖操作，都应该通过直观的操作分析模型来完成，而不需要使用菜单，也不需要跨越用户界面进行多次操作，即综合路径重定位、向上综合、向下钻取和其他操作都可以通过直观、方便的点、拉操作来完成。

在分析模型中定义的维应包含用户分析所需的所有信息，从而可以进行任意继承操作。

**11. 灵活的报表生成**

使用 OLAP 服务器及其工具，用户可以按任何想要的方式来操作、分析、综合和查看数据，这些方式包括将行、列及单元按需要依次排放。报表机制也应提供此种灵活性，报表必须能从各种可能的方面显示出从数据模型中综合出的数据和信息，充分反映数据分析模型的多维特征，并可按用户需要的方式来显示它。

**12. 不受限制的维和聚集层次**

OLAP 服务器应能在一通用分析模型中协调至少 15 个维。每一通用维应能允许有任意个用户定义的聚集，而且用户分析员可以在任意给定的综合路径上建立任意多个聚集层次(见 3.4.4 节数据立方体)。

## 3.1.3 OLAP 的基本概念

OLAP 是针对特定问题的联机数据访问和分析。通过对信息进行快速、稳定一致和交互性的存取，允许管理决策人员对数据进行深入观察。为了对 OLAP 技术有更深入的了解，这里主要介绍在 OLAP 中常用的一些基本概念。

**1. 变量**

变量是数据的实际意义，即描述数据“是什么”。例如，数据 100 本身并没有意义或者说意义未定，它可能是一个学校的学生人数，也可能是某产品的单价，还可能是某商品的销售量，等等。一般情况下，变量总是一个数值度量指标，例如，“人数”、“单价”、“销售量”等都是变量，而 100 则是变量的一个值。

**2. 维**

维是人们观察数据的特定角度。例如，企业常常关心产品销售数据随着时间推移而产生的变化情况，这时是从时间的角度来观察产品的销售，所以时间是一个维(时间维)。企业也时常关心自己的产品在不同地区的销售分布情况，这时是从地理分布的角度来观察产品的销售，所以地理分布也是一个维(地理维)。其他还有产品维、顾客维等。

**3. 维的层次**

人们观察数据的某个特定角度(即某个维)还可以存在细节程度不同的多个描述方面，

通常称这多个描述方面为维的层次。一个维往往具有多个层次。例如，描述时间维时，可以从日期、月份、季度、年等不同层次来描述，那么日期、月份、季度、年等就是时间维的层次；同样，城市、地区、国家等构成了地理维的层次。

#### 4. 维成员

维的一个取值称为该维的一个维成员。如果一个维是多层次的，那么该维的维成员是由各个不同维层次的取值组合而成的。例如，考虑时间维具有日期、月份、年这三个层次，分别在日期、月份、年上各取一个值组合起来，就得到了时间维的一个维成员，即"某年某月某日"。一个维成员并不一定在每个维层次上都要取值，例如，"某年某月"、"某月某日"、"某年"等都是时间维的维成员。对应一个数据项来说，维成员是该数据项在某维中位置的描述。例如对一个销售数据来说，时间维的维成员"某年某月某日"就表示该销售数据是"某年某月某日"的销售数据，"某年某月某日"是该销售数据在时间维上位置的描述。

#### 5. 多维数组

一个多维数组可以表示为：(维 1，维 2，…，维 $n$，变量)。例如，若日用品销售数据是按时间、地区和销售渠道组织起来的三维立方体，加上变量"销售额"，就组成了一个多维数组(地区，时间，销售渠道，销售额)，如果在此基础上再扩展一个产品维，就得到一个四维的结构，其多维数组为(产品，地区，时间，销售渠道，销售额)。

#### 6. 数据单元(单元格)

多维数组的取值称为数据单元。当多维数组的各个维都选中一个维成员时，这些维成员的组合就唯一确定了一个变量的值。那么数据单元就可以表示为：(维 1 维成员，维 2 维成员，…，维 $n$ 维成员，变量的值)。例如，在产品、地区、时间和销售渠道上各取维成员"牙膏"、"上海"、"2004 年 12 月"和"批发"，就唯一确定了变量"销售额"的一个值(假设为 100 000)，则该数据单元可表示为(牙膏，上海，2004 年 12 月，批发，100 000)。

## 3.2 OLAP 的数据模型

建立 OLAP 的基础是多维数据模型，多维数据模型的存储可以有多种不同的形式。MOLAP 和 ROLAP 是 OLAP 的两种主要形式，其中 MOLAP(Multi-dimension OLAP)是基于多维数据库的 OLAP，简称为多维 OLAP；ROLAP(Relation OLAP)是基于关系数据库的 OLAP，简称关系 OLAP。还有几种 OLAP，如 WOLAP(Web OLAP)代表网络 OLAP，HOLAP(Hybrid OLAP)代表混合 OLAP。

### 3.2.1 MOLAP 数据模型

MOLAP 数据模型是基于多维数据库的 OLAP，多维数据库(Multi Dimensional DataBase，MDDB)是以多维方式组织数据，即以维作为坐标系，采用类似于数组的形式存储

数据。多维数据库中的元素具有相同类型的数值，如销售量。例如，二维 MDDB（数组，即矩阵）的数据组织如表 3.1 所示。它代表不同产品（衣服、鞋、帽）在不同地区（北京、上海、广州）的销售量情况。

**表 3.1　MDDB（二维）数据组织**

| | 北京 | 上海 | 广州 |
|---|---|---|---|
| 衣服 | 600 | 700 | 500 |
| 鞋 | 800 | 900 | 700 |
| 帽子 | 100 | 200 | 80 |

在查询中除查询一般的"衣服在广州的销售量"外，有时查询像"衣服的总销售量"等类问题，它涉及多个数据项求和，如果采取临时进行累加计算，会使查询效率大大降低。为此，需要增加汇总数据项。在多维数据库中只需要按行或列进行求和，增加"总和"的维成员即可，如表 3.2 所示。

**表 3.2　多维数据库中含综合数据的数据组织**

| | 北京 | 上海 | 广州 | 总和 |
|---|---|---|---|---|
| 衣服 | 600 | 700 | 500 | 1800 |
| 鞋 | 800 | 900 | 700 | 2400 |
| 帽子 | 100 | 200 | 80 | 380 |
| 总和 | 1500 | 1800 | 1280 | 4580 |

MDDB 的数据组织形式不同于关系数据库的组织形式，关系数据库是以"属性—元组（记录）"形式组织数据。对表 3.1 中的数据按关系数据库组织，数据如表 3.3 所示。

**表 3.3　关系数据库 RDBMS 数据组织**

| 产品名 | 地区 | 销售量 | 产品名 | 地区 | 销售量 |
|---|---|---|---|---|---|
| 衣服 | 北京 | 600 | 鞋 | 广州 | 700 |
| 衣服 | 上海 | 700 | 帽子 | 北京 | 100 |
| 衣服 | 广州 | 500 | 帽子 | 上海 | 200 |
| 鞋 | 北京 | 800 | 帽子 | 广州 | 80 |
| 鞋 | 上海 | 900 | | | |

可见，多维数据库 MDDB 比关系数据库表达更清晰且占用的存储少。在关系数据库中增加综合数据项，如表 3.4 所示。这些综合数据项一般在建立数据库的同时计算出来。这样在查询时，不必临时进行计算，提高了查询效率。对于多维数据库的综合数据项明显比关系数据库的综合项更有效果。

表 3.4　关系数据库中综合数据的数据组织

| 产品名 | 地区 | 销售量 | 产品名 | 地区 | 销售量 |
|---|---|---|---|---|---|
| 衣服 | 北京 | 600 | 鞋 | 广州 | 700 |
| 衣服 | 上海 | 700 | 鞋 | 总和 | 2400 |
| 衣服 | 广州 | 500 | 帽子 | 北京 | 100 |
| 衣服 | 总和 | 1800 | 帽子 | 上海 | 200 |
| 鞋 | 北京 | 800 | 帽子 | 广州 | 80 |
| 鞋 | 上海 | 900 | 帽子 | 总和 | 380 |

### 3.2.2　ROLAP 数据模型

ROLAP 是基于关系数据库的 OLAP,如表 3.3 所示。它是一个平面结构,用关系数据库表示多维数据时,采用星型模型,即用两类表,一类是事实表,存储事实的实际值,如销售量;另一类是维表,对每一个维来说,至少有一个表来存储该维的描述信息,如产品的名称、分类等。星型模型完全用二维关系表示了数据的多维观念。

通过关系数据库实现多维查询时,通过维表的主码对事实表和每一个维表做连接操作,一次查询就可以得到数据的具体值以及对数据的多维描述(即对应的各维上的维成员)。但是,因为对每个维都需要进行一次连接操作,所以系统的性能就成了 ROLAP 实现的最大的一个问题,特别是当维数增加和事实表增大时,必须采用有效的查询优化技术(特别是表连接策略),利用各种索引技术来提高系统的性能。

当存在多层次的复杂维时,需要采用"雪花模型",用多张表来描述一个复杂维。对于存在综合数据时,需要建立汇总事实表,采用"星网模型"来描述。

### 3.2.3　MOLAP 与 ROLAP 的比较

MOLAP 通过多维数据库引擎从关系数据库 DB 和数据仓库 DW 中提取数据,将各种数据组织成多维数据库,存放到 MDDB 中,而且将自动建立索引并进行预综合(见 3.4.4 节)来提高查询存取性能,如图 3.1 所示。

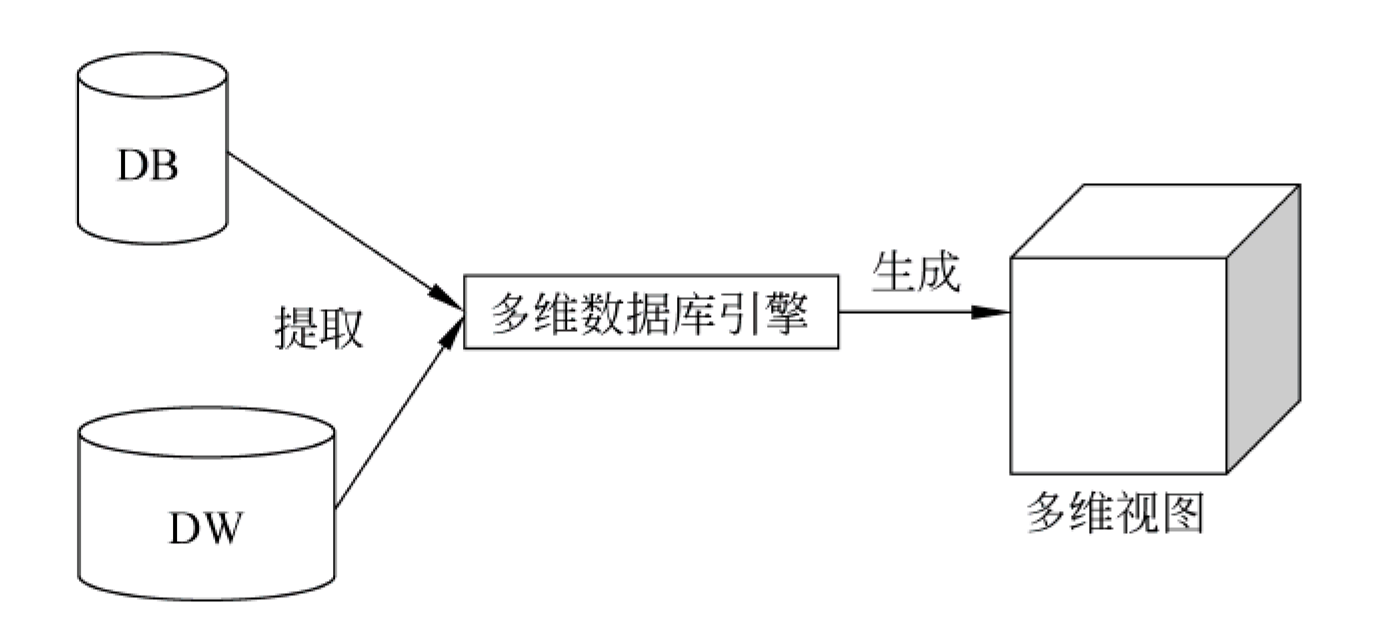



图 3.1　MOLAP 结构

ROLAP 从关系数据库 DB 和数据仓库 DW 中提取数据,按关系 OLAP(ROLAP)的数据组织存放在关系数据库服务器(RDBMS 服务器)中。最终用户的多维分析请求,通过

ROLAP 服务器的多维分析(OLAP)引擎动态翻译成 SQL 请求,将查询结果经多维处理(将关系表达式转换成多维视图)返回用户,如图 3.2 所示。

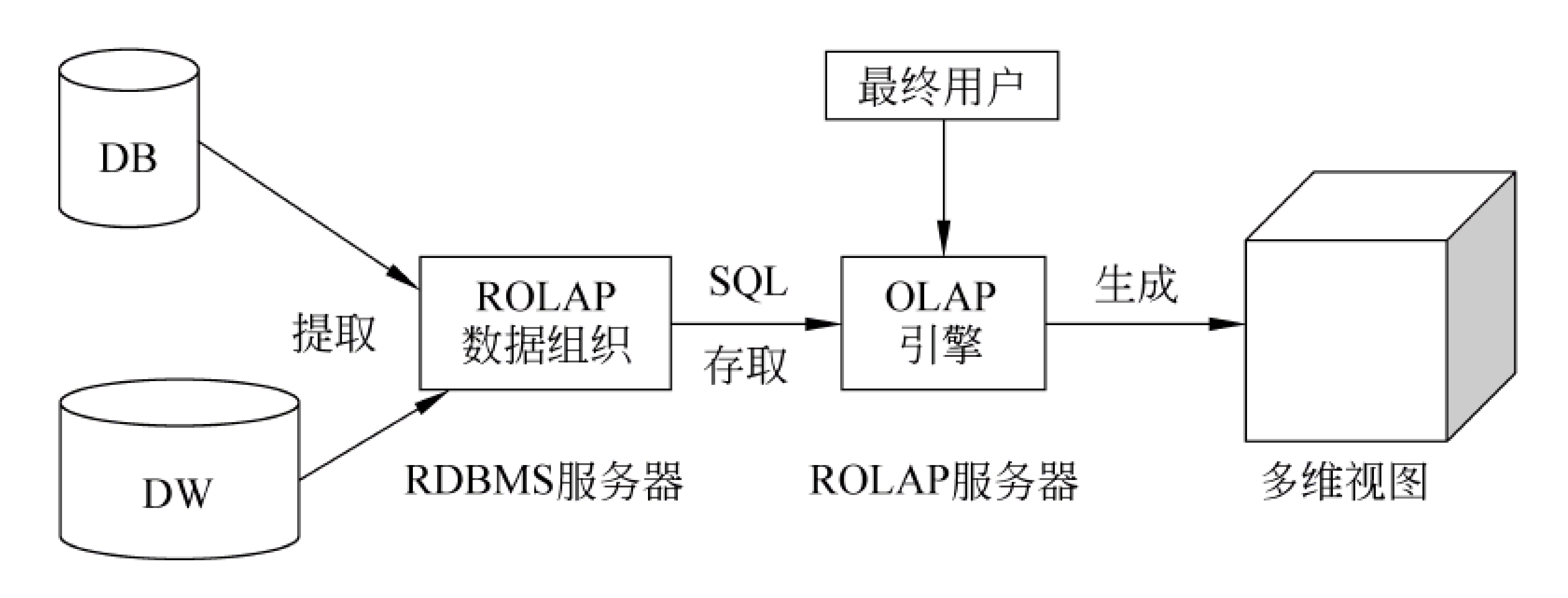



图 3.2 ROLAP 结构

虽然这两种技术都满足了 OLAP 数据处理的一般过程,即数据装入、汇总、建索引和提供使用,但 MOLAP 要比 ROLAP 简明一些,MOLAP 的索引及数据综合可以自动进行。然而 ROLAP 的实现较为复杂,但灵活性较好,用户可以动态实现统计或计算方式。

下面详细深入分析 MOLAP 与 ROLAP。

### 1. 数据存取速度

ROLAP 的多维数据是以星型模型等关系数据库(平面形式)存储,并不直接体现“超立方体”形式。在接收客户 OLAP 请求时,ROLAP 服务器需要将 SQL 语句转化为多维存储语句,并利用连接运算临时“拼合”出多维数据立方体。因此,ROLAP 的响应时间较长。

目前,关系型数据库已经对 OLAP 做了很多优化,包括并行存储、并行查询、并行数据管理、基于成本的查询优化、位图索引、SQL 的 OLAP 扩展等,大大提高了 ROLAP 的速度。

MOLAP 是专为 OLAP 所设计的,能够自动地建立索引,并且有良好的预计算能力,能够使用多维查询语句访问数据立方体,因此 MOLAP 在数据存储速度上性能好,响应速度快。

### 2. 数据存储的容量

ROLAP 使用的传统关系数据库的存储方法,在存储容量上基本没有限制。但是,需要指出的是,在 ROLAP 中为了提高分析响应速度,常常构造大量的中间表(如综合表),这些中间表带来了大量的冗余数据。

MOLAP 通常采用多平面叠加成立体的方式存放数据,(这样访问速度快),由于受操作系统平台中文件大小的限制,当数据量超过操作系统最大文件长度时,需要进行数据分割。随着数量的增大,多维数据库进行的预运算结果将占用巨量的空间,此时可能会导致“数据爆炸”的现象。因此,多维数据库的数据量级难以达到太大的字节级。

### 3. 多维计算的能力

MOLAP 能够支持高性能的决策支持计算,包括复杂的跨维计算、行级的计算,而在 ROLAP 中,SQL 无法完成部分计算,并且 ROLAP 无法完成多行的计算和维之间的计算。

最近发展起来的多维数据分析语言 MDX 能更有效地进行多维数据分析(见 3.4.5 节)。

#### 4. 维度变化的适应性

MOLAP需要在建立多维数据库前确定各个维度以及维度的层次关系。在多维数据库建立之后，如果要增加新的维度，则多维数据库通常需要重新建立。新增维度数据会剧烈增加。而ROLAP增加一个维度，只是增加一张维表并修改事实表，系统中其他维表不需要修改，因此ROLAP对于维表的变更有很好的适应性。

#### 5. 数据变化的适应性

由于MOLAP通过预综合处理来提高速度，当数据频繁地变化时，MOLAP需要进行大量的重新计算，甚至重新建立索引乃至重构多维数据库。在ROLAP中，预综合处理通常由设计者根据需求制定，因此灵活性较好，对于数据变化的适应性强。

#### 6. 软硬件平台的适应性

关系数据库已经在众多的软硬件平台上成功地运行，即ROLAP对软硬件平台的适应性很好，而MOLAP相对较差。

#### 7. 元数据管理

元数据是OLAP和数据仓库的核心数据，OLAP的元数据包括层次关系、计算转化信息、报表中的数据项描述、安全存取控制、数据更新、数据源和预计算综合表等，目前在元数据的管理上，MOLAP和ROLAP都没有成形的标准，MOLAP产品将元数据作为其内在数据，而ROLAP产品将元数据作为应用开发的一部分，由设计者来定义和处理。

MOLAP和ROLAP在技术上各有优缺点。MOLAP以多维数据库为核心，在数据存储和综合上有明显的优势，但它不适应太大的数据存储，特别是对有大量稀疏数据的存储将会浪费大量的存储空间。ROLAP以RDBMS为基础，利用成熟的技术为用户的使用和管理带来方便。

MOLAP和ROLAP在数据存储、技术和特性方面的比较如表3.5所示。

**表3.5　MOLAP和ROLAP的比较**

| | 数据存储 | 技　术 | 特　征 |
|---|---|---|---|
| MOLAP | 详细数据用关系表存储在数据仓库中；<br>各种汇总数据保存在多维数据库中；<br>从数据仓库中询问详细数据，从多维数据库中询问汇总数据 | 由MOLAP引擎创建；<br>预先建立数据立方体；<br>多维视图存储在陈列中，而不是表格中；<br>可以高速检索矩阵数据；<br>利用稀疏矩阵技术来管理汇总的稀疏数据 | 询问响应速度快；<br>能轻松适应多维分析；<br>有广泛的下钻和多层次/多视角的查询能力 |
| ROLAP | 全部数据以关系表存储在数据仓库中；<br>可获得细节的和综合汇总的数据；<br>有非常大的数据容量；<br>从数据仓库中询问所有的数据 | 使用复杂SQL从数据仓库中获取数据；<br>ROLAP引擎在分析中创建数据立方体；<br>表示层能够表示多维的视图 | 在复杂分析功能上有局限性，需要采用优化的OLAP；<br>向下钻取较容易，但是跨维向下钻取比较困难 |

### 3.2.4 HOLAP 数据模型

HOLAP(Hybrid OLAP),即混合 OLAP 介于 MOLAP 和 ROLAP 之间。在 HOLAP 中,对于最常用的维度和维层次,使用多维数据库来存储,对于用户不常用的维度和数据,采用 ROLAP 星型结构来存储。当用户询问不常用数据时,HOLAP 将会把简化的多维数据库和星型结构进行拼合,从而得到完整的多维数据。

在 HOLAP 的多维数据库中的数据维度少于 MOLAP 中的维度库,数据存储容量也少于 MOLAP 方式。但是,HOLAP 在数据存取速度上又低于 MOLAP。

## 3.3 多维数据的显示

### 3.3.1 多维数据显示方法

多维数据一般采用多维数据库(MDDB)和关系数据库(RDBMS)两种方式存储。多维数据的显示只能在平面上展现出来。对于二维数据采用多维数据库形式显示时,如表 3.1 所示。二维数据采用关系数据库形式显示时,如表 3.3 所示。若增加一维时间维,仍然可以显示出来,如表 3.6 所示。

表 3.6 三维数据的关系数据库显示

| 产品名 | 地区 | 时间 | 销售量 |
|---|---|---|---|
| 衣服 | 北京 | 1 月 | 100 |
| 衣服 | 北京 | 2 月 | 200 |
| 衣服 | 北京 | 3 月 | 300 |
| 衣服 | 上海 | 1 月 | 200 |
| 衣服 | 上海 | 2 月 | 300 |
| 衣服 | 上海 | 3 月 | 400 |
| 衣服 | 广州 | 1 月 | 150 |
| 衣服 | 广州 | 2 月 | 250 |
| 衣服 | 广州 | 3 月 | 300 |
| 鞋 | 北京 | 1 月 | 150 |
| 鞋 | 北京 | 2 月 | 300 |
| 鞋 | 北京 | 3 月 | 350 |
| 鞋 | 上海 | 1 月 | 200 |
| 鞋 | 上海 | 2 月 | 300 |
| 鞋 | 上海 | 3 月 | 400 |
| 鞋 | 广州 | 1 月 | 150 |
| 鞋 | 广州 | 2 月 | 250 |
| 鞋 | 广州 | 3 月 | 300 |
| ⋮ | ⋮ | ⋮ | ⋮ |

用关系数据库可以显示更多维的数据，即用星型模型的事实表形式显示。但是，用事实表显示多维数据时，重复数据很多，也显得很烦琐。

用多维数据库显示时，虽然不能同时显示三维以上数据，由于显示的数据很精炼，因此仍然用多维数据库的方式来显示多维数据。一般在多维数据库中，固定一些维成员，重点显示两维的数据。如在表3.6三维数据中，固定地区维是"北京地区"时的两维数据的显示如表3.7所示。

**表3.7 北京地区销售情况表**

| 北京地区 | 1月 | 2月 | 3月 |
|---|---|---|---|
| 衣服 | 100 | 200 | 300 |
| 鞋子 | 150 | 300 | 350 |
| ⋮ | ⋮ | ⋮ | ⋮ |

## 3.3.2 多维类型结构

为了有效地表示多维数据，E. Thomsen 引入了多维类型结构(MTS)。有些专家称之为多维域结构(MDS)。表示方法是：每一个维度用一条线段来表示。维度中的每一个成员都用线段上的一个单位区间来表示。例如，用三个线段分别表示时间、产品和指标三个维的多维类型结构如图3.3所示。

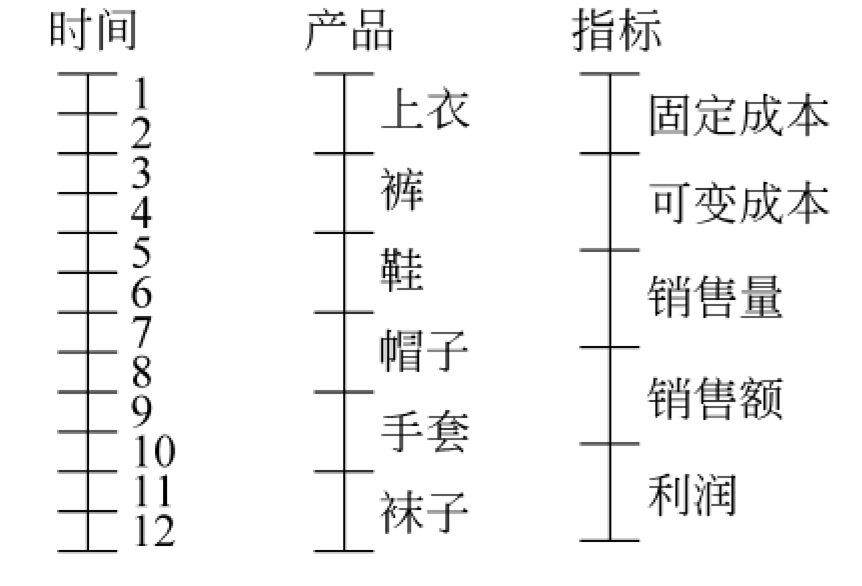


图3.3 三维MTS例

在图3.3多维类型结构(MTS)中，指定时间维成员是3月，产品维成员是鞋，指标维成员是销售量，这样它代表了三维数据的一个空间数据点，如图3.4所示。

在MTS中，在原有多维数据中增加一个维是很容易的，例如在图3.3的三维中增加一个商店维，这时需要增加一条线段表示商店维，如图3.5所示。

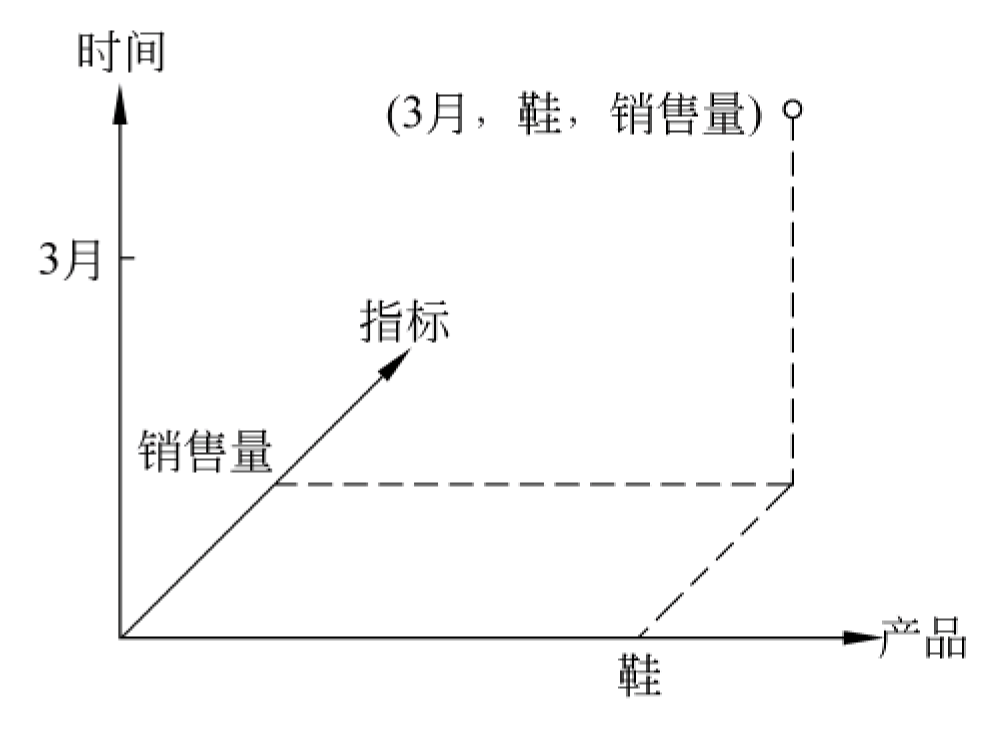


图3.4 多维类型结构中的空间数据点

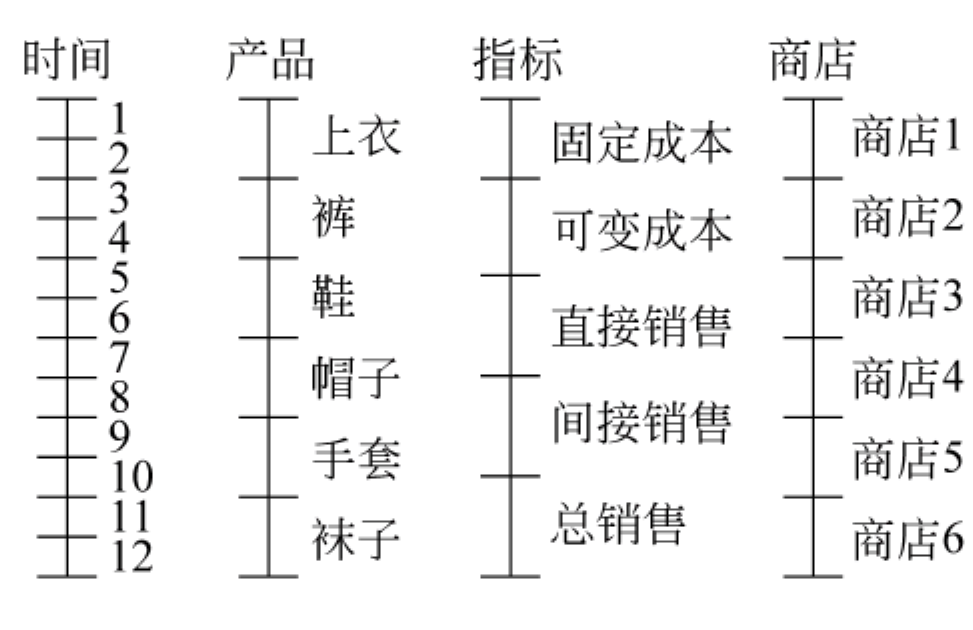


图3.5 四维MTS例

## 3.3.3 多维数据的分析视图

在平面的屏幕上显示多维数据，是利用行、列和页面三个显示组来表示的。例如，对上

例的四维 MTS 实例，在页面上选定商店维度的“商店 3”，在行中选定时间维的“1 月、2 月、3 月”共 3 个成员，在列中选定产品维中的“上衣、裤、帽子”三个成员，以及指标维中的“固定成本、直接销售”两个成员。该四维数据的显示如图 3.6 所示。

| 商店3<br>(页面) | 上衣 | | 裤 | | 帽子 | |
|---|---|---|---|---|---|---|
| | 直接销售 | 固定成本 | 直接销售 | 固定成本 | 直接销售 | 固定成本 |
| 1月 | 450 | 350 | 550 | 450 | 500 | 400 |
| 2月 | 380 | 280 | 460 | 360 | 400 | 320 |
| 3月 | 400 | 310 | 480 | 410 | 450 | 400 |

图 3.6　四维数据的显示

对于更多维度的数据显示，需要选择维度及其成员分布在行或者列中。在页面上可以选定多个维度，但每个维度只能显示一个成员。在行或者列中一般只选择两个维，每个维都可以多个成员。例如，对 6 个维度数据，它的 MTS 如图 3.7 所示。

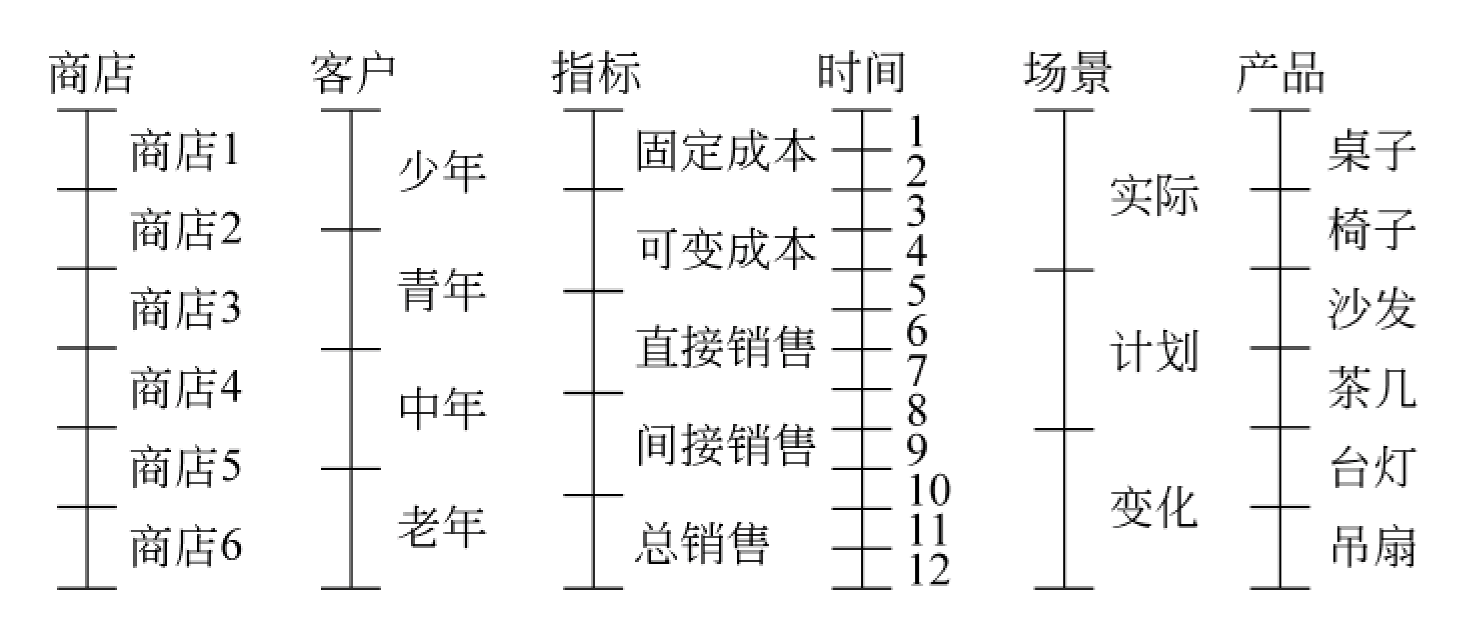



图 3.7　六维 MTS 实例

对以上六维数据中，设定页面维度为商店的成员是“商店 3”，客户维度成员是“老年”。行维度含时间维和产品维共 2 个维度，其中时间维中成员为“1 月、2 月、3 月”。产品维中成员为“桌子、台灯”。列维度含指标维和场景维共 2 个维度，其中指标维中成员为“直接销售、间接销售、总销售”。场景维中成员为“实际、计划”。具体的显示数据如图 3.8 所示。

| 商店3，老年<br>(页面) | | 直接销售 | | 间接销售 | | 总销售 | |
|---|---|---|---|---|---|---|---|
| | | 实际 | 计划 | 实际 | 计划 | 实际 | 计划 |
| 1月 | 桌子 | 250 | 300 | 125 | 150 | 375 | 450 |
| | 台灯 | 265 | 320 | 133 | 160 | 400 | 480 |
| 2月 | 桌子 | 333 | 400 | 167 | 200 | 500 | 600 |
| | 台灯 | 283 | 340 | 142 | 170 | 425 | 510 |
| 3月 | 桌子 | 350 | 420 | 175 | 210 | 525 | 630 |
| | 台灯 | 250 | 300 | 125 | 150 | 375 | 450 |

图 3.8　六维数据的显示

由于整个屏幕的空间是有限的，将维度嵌套在行或者列中相对于放在页维度中会占据更多的屏幕空间。用于显示维度的空间越多，那么用于显示数据的空间就会越少。随着显示数据空间的减少，为了查看同样的数据，就需要做更多的卷屏操作。卷屏操作的增加也加大了理解正在寻找的数据的难度。一些经验规则有：

（1）将维度尽量放在页中，除非确定需要同时看到一个维度的多个成员。让屏幕上的信息尽量相关。

（2）当维度嵌套在行或者列中时，考虑到垂直空间比水平空间更为有用，所以将维度嵌套在列中比嵌套在行中要好。一个经典的显示方法就是在行上有 1 个维度，而在列上嵌套 1～3 个维度，而其他的维度则放在页中，如图 3.6 所示。

（3）在决定数据的屏幕显示方式之前，应该首先弄清楚需要查找和分析比较的内容。例如，如果需要比较某个产品和某类客户在商品和时间上的实际成本情况，就可以将产品和客户放在页面维度中，而在屏幕上则可以按商店和时间来显示实际成本，如图 3.9 所示。

页面维度：

产品维成员“鞋”，指标维成员“成本”，场景维成员“实际”，客户维成员“青年”。

| | 1月 | 2月 | 3月 | 4月 |
|---|---|---|---|---|
| 商店1 | 125 | 170 | 157 | 114 |
| 商店2 | 200 | 195 | 129 | 157 |
| 商店3 | 136 | 158 | 132 | 144 |

图 3.9　按照商店和时间比较成本的数据组织

## 3.4　OALP 的多维数据分析

### 3.4.1　多维数据分析的基本操作

OLAP 的目的是为管理决策人员通过一种灵活的多维数据分析手段，提供辅助决策信息。基本的多维数据分析操作包括切片、切块、旋转、钻取等。通常把在多维数据分析中加入数据分析模型和商业分析模型称为广义 OLAP。

随着 OLAP 的深入发展，出现了多维数据聚集计算的数据立方体和多维数据分析的 MDX 语言。

**1. 切片**

选定多维数组的一个二维子集的操作叫做切片（Slice），即选定多维数组（维 1，维 2，…，维 $n$，变量）中的两个维：如维 $i$ 和维 $j$，在这两个维上取某一区间或任意维成员，而将其余的维都取定一个维成员，则得到的就是多维数组在维 $i$ 和维 $j$ 上的一个二维子集，称这个二维子集为多维数组在维 $i$ 和维 $j$ 上的一个切片，表示为（维 $i$，维 $j$，变量）。

切片就是在某两个维上取一定区间的维成员或全部维成员，而在其余的维上选定一个维成员的操作。这里可以得出两点共识：

维是观察数据的角度，那么切片的作用或结果就是舍弃一些观察角度，使人们能在两个维上集中观察数据。因为人的空间想象能力毕竟有限，一般很难想象四维以上的空间结构。所以对于维数较多的多维数据空间，数据切片是十分有意义的。

图 3.10 所示为一个按产品维、地区维和时间维组织起来的产品销售数据，用三维数组表示为（地区，时间，产品，销售额）。如果在地区维上选定一个维成员（设为“上海”），就得到了在地区维上的一个切片（关于“时间”和“产品”的切片）；在产品维上选定一个维成员（设为

“电视机”)，就得到了在产品维上的一个切片(关于“时间”和“地区”的切片)。显然，切片的数目取决于每个维上维成员的个数。

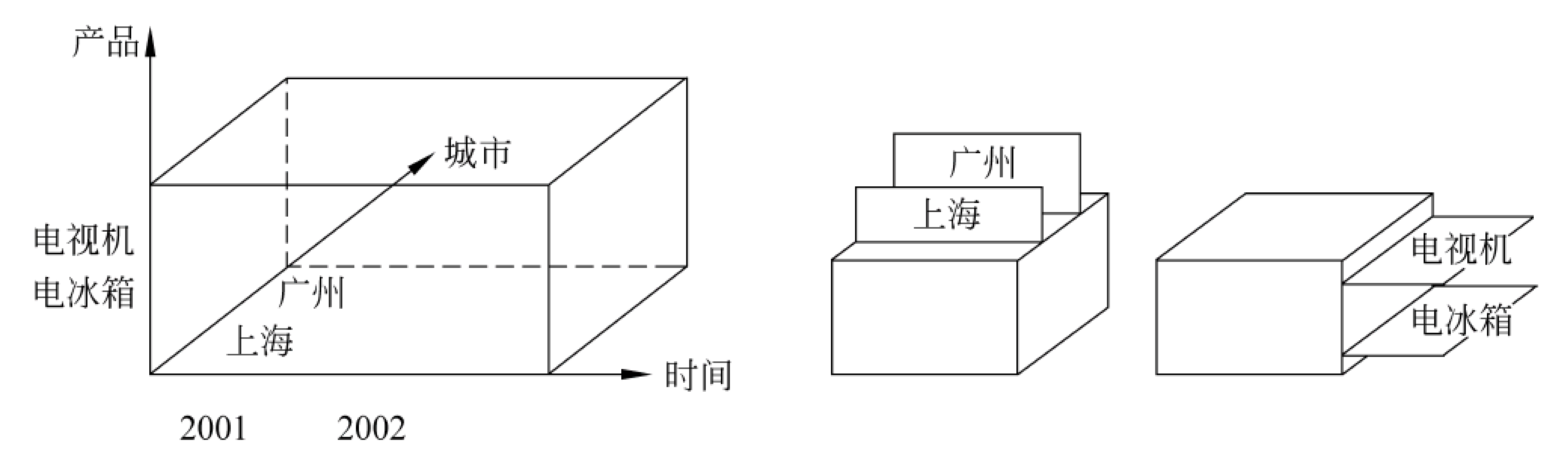



图 3.10　三维数据切片

**2. 切块**

切块(Dice)有如下两种情况。

(1) 在多维数组的某一个维上选定某一区间的维成员的操作

切块可以看成是在切片的基础上确定某一个维成员的区间得到的片段，也即由多个切片叠合起来的。对于时间维的切片(时间取一个确定值)，如果将时间维上的取值设定为一个区间(例如取“2001 年至 2005 年”)，就得到一个数据切块，它可以看成由 2001 年至 2005 年 5 个切片叠合而成的。

(2) 选定多维数组的一个三维子集的操作

在多维数组(维 1，维 2，…，维 $n$，变量)中选定 3 个维，维 $i$、维 $j$、维 $k$，在这 3 个维上分别取一个区间，或任意维成员，而其他维都取定一个维成员。例如在 3 维数组(地区、时间、产品、销售额)中地区维取上海与广州两个维成员，产品维取电视机、电冰箱两个维成员，时间维取 2003 到 2005 的区间(三个维成员)组成三维立方体，如图 3.11 所示。

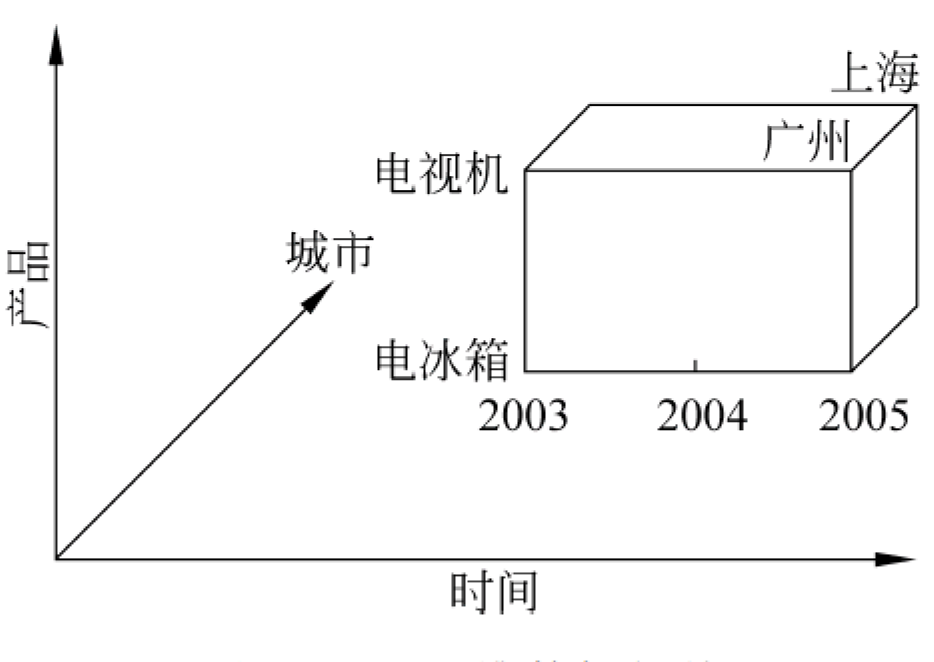



图 3.11　三维数据切块

**3. 钻取**

钻取(Drill)分为向下钻取(drill down )和向上钻取(drill up )操作。向下钻取是使用户在多层数据中能通过导航信息而获得更多的细节性数据，而向上钻取获取概括性的数据。例如，2009 年各部门销售收入如表 3.8 所示。

**表 3.8　部门销售数据**

| 部门 | 销售 | 部门 | 销售 |
|---|---|---|---|
| 部门 1 | 900 | 部门 3 | 800 |
| 部门 2 | 600 | | |

在时间维进行下钻(drill down)操作，获得新表 3.9。

**表 3.9　部门销售下钻数据**

| | 2009 年 | | | |
|---|---|---|---|---|
| 部门 | 1 季度 | 2 季度 | 3 季度 | 4 季度 |
| 部门 1 | 200 | 200 | 350 | 150 |
| 部门 2 | 250 | 50 | 150 | 150 |
| 部门 3 | 200 | 150 | 180 | 270 |

相反的操作为上钻(drill up)。钻取的深度与维所划分的层次相对应。

**4. 旋转**

通过旋转(Pivot)可以得到不同视角的数据。旋转操作相当于平面数据将坐标轴旋转。例如,旋转可能包含了交换行和列,或是把某一个行维移到列维中去,或是把页面显示中的一个维和页面外的维进行交换(令其成为新的行或列中的一个),如图 3.12 所示。

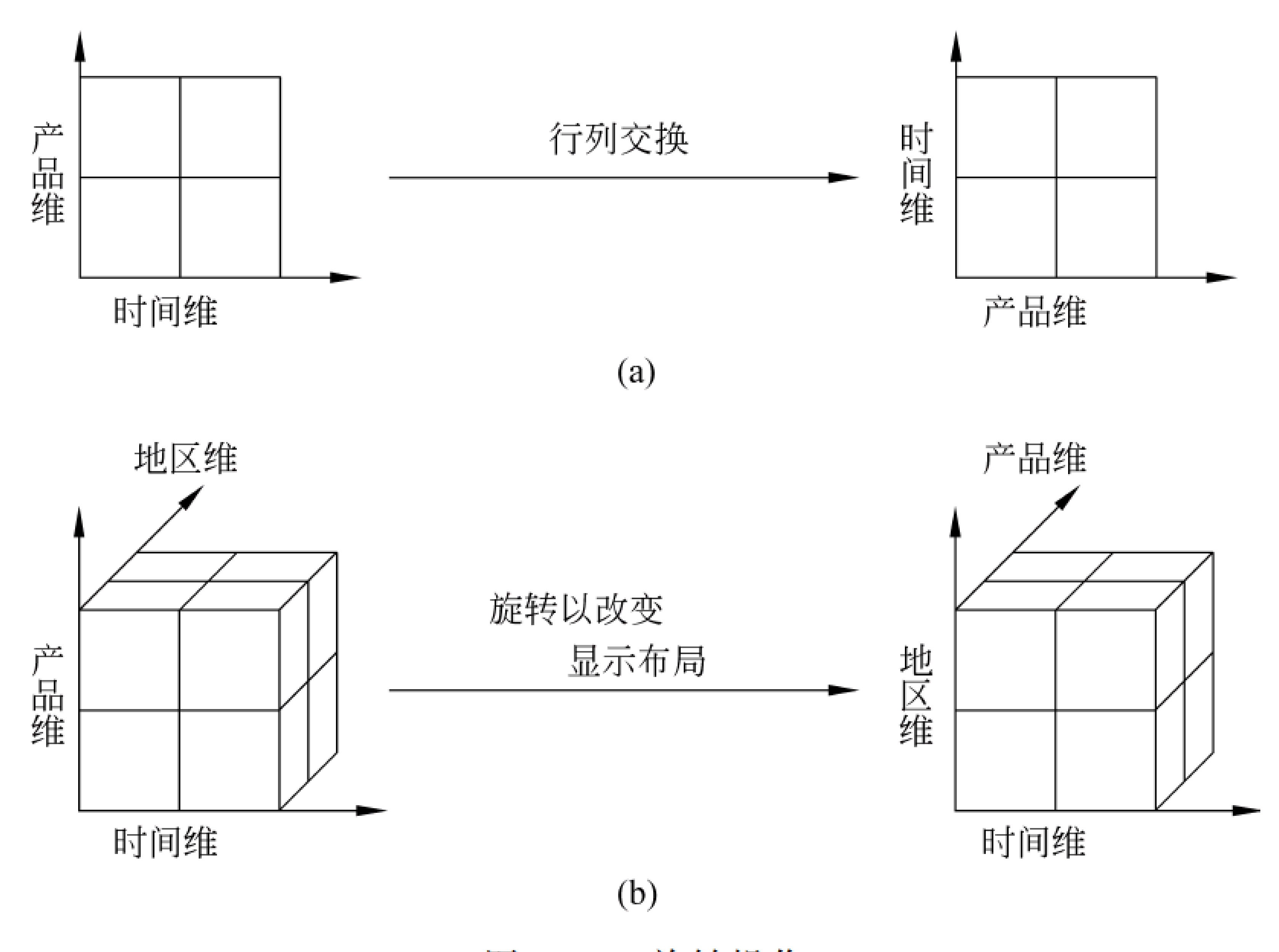



图 3.12　旋转操作

图 3.12(a)是把一个横向为时间、纵向为产品的报表旋转成为横向为产品、纵向为时间的报表。

图 3.12(b)是把一个横向为时间、纵向为产品的报表旋转成一个横向仍为时间而纵向为地区的报表。

## 3.4.2　多维数据分析实例

**1. 切片**

为了对广东省全省营业税和个人所得税在 2006—2007 两年的纳税情况进行全面了解,需要对全省税收数据按城市进行切片显示,部分城市数据如表 3.10 所示。

表 3.10　广东省各市营业税和个人所得税表　　　单位：亿元

| | 2006 年营业税 | 2006 年所得税 | 2007 年营业税 | 2007 年所得税 |
|---|---|---|---|---|
| 广州市 | 199 | 96 | 231 | 122 |
| 东莞市 | 53.4 | 25.4 | 70.3 | 31.6 |
| 珠海市 | 23.9 | 9.1 | 34.9 | 13.9 |
| 佛山市 | 55.7 | 29.3 | 72.5 | 34.4 |

由表 3.10 中数据可知，广州营业税增加 32.8 亿元，增长率为 16.5%。广州个人所得税增加 25.6 亿元，增长率为 26.7%。东莞营业税增加 16.9 亿元，增长率为 31.68%。东莞个人所得税增加 6.2 亿元，增长率为 24.6%。

对营业税而言，增长量最大的城市是广州，增加速度较快的城市是东莞(31.68%)。

**2. 向下钻取**

为了更深入分析东莞市的各行业的营业税情况，需要对东莞营业税数据下钻分析。2006、2007 两年部分行业的纳税情况如表 3.11 所示。

表 3.11　东莞市各行业的营业税表　　　单位：百万元

| | 2006 年营业税 | 2007 年营业税 |
|---|---|---|
| 农、林、牧、渔业 | 15 | 10 |
| 房地产业 | 1204 | 1510 |
| 制造业 | 85.5 | 112.8 |
| 餐饮业 | 327.9 | 363.8 |
| 金融业 | 475.7 | 698.1 |
| 采矿业 | 0.028 | 0.026 |

由表 3.11 中数据可知，东莞市农、林、牧、渔业 2007 年下降了 5.5 百万元，下降率为 35.2%。采矿业下降 0.02 百万元，下降率为 10.5%。房地产业增加 306.5 百万元，增加率为 21%。金融业增加 222.4 百万元，增加率为 46.8%。对这四种行业的增减率有更直观的表示，用直方图表示，如图 3.13 所示。

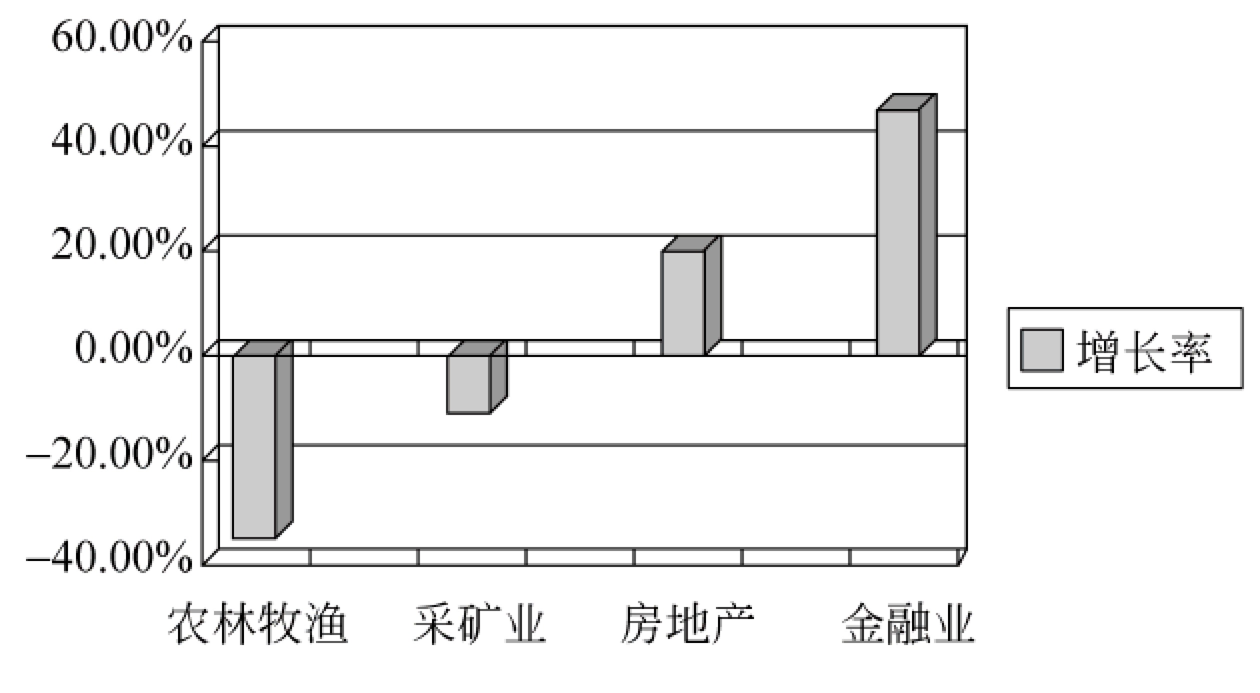



图 3.13　东莞市四个行业营业税增减率的直方图

### 3. 数据分析

(1) 宏观分析

从表 3.10 中的数据可以宏观地看出，东莞市的营业税增长很突出，在广东省各市中名列前位。

(2) 深入分析

根据表 3.11 中的行业数据进行深入分析时发现，东莞的农、林、牧、渔业下降明显，采矿业也下降，而房地产业增长明显，金融业增长突出。通过调查得出，原因是随着经济的发展，东莞的外来合资企业越来越多，本地农民很多把地卖了或者租出去建厂房然后收租金，造成农林渔营业税下降，东莞市近年逐步实现产业转移，由农业更多地转向制造业和加工业。

从总体看，东莞的房地产业和金融业税收的上升，掩盖了农、林、牧、渔业税收的下降。对领导来说，这是好事，要继续支持？还是不合理，需要调整？这就需要领导做出正确的决策。

## 3.4.3 广义 OLAP 功能

OLAP 的切片、切块、旋转与钻取等基本操作是最基本的展示数据、获取数据信息的手段。从广义上讲，任何能够有助于辅助用户理解数据的技术或者操作都可以作为 OLAP 功能，这些有别于基本 OLAP 的功能称为广义 OLAP 功能。

### 1. 基本代理操作

“代理”是一些智能性代理，当系统处于某种特殊状态时提醒分析员。

(1) 示警报告

定义一些条件，一旦条件满足，系统会提醒分析员去做分析。例如每日报告完成或月订货完成等通知分析员做分析。

(2) 时间报告

按日历和时钟提醒分析员。

(3) 异常报告

当超出边界条件时提醒分析员。例如销售情况已超出预定义阈值的上限或下限时提醒分析员。

### 2. 数据分析模型

E. F. Codd 认为，以前的数据分析主要集中在静态数据值的相互比较上。有了 OLAP 后，可以进行动态数据分析，需要建立企业数据模型。Codd 将数据分析模型分为四类：绝对模型(Categorical Model)、解释模型(Exegetical Model)、思考模型(Contemplative Model)和公式化(Formulaic Model)。

(1) 绝对模型

它属于静态数据分析，通过比较历史数据值或行为来描述过去发生的事实。该模型查询比较简单，综合路径是预先定义好的，用户交互少。

(2) 解释模型

它也属于静态数据分析，分析人员利用系统已有的多层次的综合路径层层细化(进行向

下钻取操作)，找出事实发生的原因。

(3) 思考模型

它属于动态数据分析，旨在说明在一维或多维上引入一组具体变量或参数后将会发生什么。分析人员在引入确定的变量或公式关系时，须创建大量的综合路径。

(4) 公式模型

它的动态数据分析能力更强，该模型表示在多个维上，需要引入哪些变量或参数，以及引入后所产生的结果。

下面通过一个实例进行说明。

一家百货公司在建立了自己的数据仓库之后，希望构造一个 OLAP 系统辅助决策。决策者最关心的一个问题是如何最大限度地扩大商品的销售量，因而他希望能尽可能地找出与销售量相关的因素，从而采取相应的促销手段。但是他能获得多大的帮助需要取决于采用何种分析模型。

绝对模型只能对历史数据进行比较，并且利用回归分析等一些分析方法得出趋势信息。它能回答诸如“某种商品今年的销售情况与以往相比有怎样的变化？今后的趋势怎样?”此类问题。

解释模型能够在当前多维视图的基础上找出事件发生的原因。例如，该公司按时间、地区、商品及销售渠道建立了多维数据库，假设今年销售量下降，那么解释模型应当能找出原因，即销售量下降与时间、地区、商品及销售渠道四者中的何种因素有关。

思考模型可以在决策者的参与下，找出关键变量。例如该公司决策者为了了解某商品的销售量是否与顾客的年龄有关，引入了新变量一年龄，即在当前的多维视图上增加了顾客的年龄维。解释模型就能分析出年龄的引入是否必要，即商品销售与顾客年龄有关或无关。

公式模型自动完成上述各种变量的引入和分析，从而最终找出与销量有关的全部因素，并给出了引入各变量后的结果。

可以看出，这四种模型一个比一个深入，从描述基本事实到寻找原因，从代入变量值进行预测到寻找关键变量。

Codd 认为 OLAP 是因企业动态分析而产生的，其功能是创建、操作、激活及综合来自解释模型、思考模型及公式化模型中的信息。它可以识别变量间新的或不可预测的关联，通过创建大量的维(综合路径)及指出维间计算条件、表达式来处理大量数据，获得辅助决策信息。

### 3. 商业分析模型

利用数据仓库中的数据进行商业分析需要建立一系列模型，用于提高决策支持能力。

具体的商业分析模型有：

(1) 分销渠道的分析模型

通过客户、渠道、产品或服务三者之间的关系，了解客户的购买行为、客户和渠道对业务收入的贡献、哪些客户比较喜好由什么渠道在何时和银行打交道、目前的分销渠道的服务能力如何、需要增加哪些分销渠道才能达到预期的服务水平。

为此，银行需要建立客户购买倾向模型和渠道喜好模型等。

(2) 客户利润贡献度模型

通过该模型能了解每一位客户对银行的总利润贡献度，银行可以依客户的利润贡献度

安排合适的分销渠道提供服务和销售,知道哪些利润高的客户需要留住,采用什么方法留住客户,交叉销售改善客户的利润贡献度,哪些客户应该争取,完成个性化服务。另外,银行可以模拟和预测新产品对银行的利润贡献度,或者新政策对银行将产生什么样的财务影响,或者客户流失或留住对银行整体利润的影响。

(3) 客户关系(信用)优化模型

银行对客户的每一笔交易中,知道客户需要什么产品或服务,例如,定期存款是希望退休养老使用、申请信用卡需要现金消费、询问放贷利息、需要住房贷款等,这些都是银行提供产品或服务最好的时机。银行需要将账号每天发生的交易明细,以实时或定时方式加载到数据仓库中,校对客户行为的变化。当发生上述变化时,通过模型计算,主动地与客户沟通并进行交叉销售,达到留住客户和增加利润的目标。

(4) 风险评估模型

模拟风险和利润间的关系,建立风险评估的数学模型,在满足高利润、低风险客户需求的前提下,达到银行收益的极大化。

银行通过以上模型实现以客户为中心的数据仓库决策支持系统,才能真正实现个性化服务,提高银行竞争优势。

### 3.4.4 数据立方体

#### 1. 概述

1996 年,Jim Gray 等首次提出了数据立方体(Data Cube)的概念,数据立方体是实现多维数据查询与分析的一种重要手段。实质上,数据立方体就是数据仓库结构图(见图 2.1)中的综合数据层(轻度和高度)。从此,基于数据立方体的生成方法一直是 OLAP 和数据仓库领域研究者所关注的热点问题。

多维数据集的属性分为维属性和度量属性。维属性是观察数据对象的角度,而度量属性则反映数据对象的特征。对于多维数据分析而言,本质上是沿着不同的维度进行数据获取的过程。在数据立方体中,不同维度组合构成了不同的子立方体,不同维值的组合及其对应的度量值构成相应的对于不同的查询和分析。因此,数据立方体的构建和维护等计算方法成为了多维数据分析研究的关键问题。

OLAP 和数据仓库通常预先计算好不同细节层次和不同维属性集合上的聚集,并把聚集的结果存储到物理磁盘上(称为物化)。把所有可能的聚集(即全聚集)都计算出来,可以得到最快的系统查询响应时间,即使不管计算聚集所花费的 CPU 处理时间, 只是随着维数的增加, 这样做就有可能导致数据爆炸。

数据立方体是在所有可能组合的维上进行分组聚集运算(group by 操作)的总和,聚集函数有:sum()、count()、average() 等。数据立方体中的每一个元组(立方体的度量属性)被称为该立方体上的格(cell),每个格在 $n$ 个维属性上有相应的值,其中,在未参与 group by 操作的维属性上具有 All 值(用 * 表示),而在参与 group by 操作的维属性具有非 All 值。

例如,对于一个具有三个维属性 $A$、$B$、$C$ 和一个度量属性 $M$ 的数据集 $R(A,B,C,M)$,其对应的数据立方体是在维属性集$\{\ \}$,$\{A\}$,$\{B\}$,$\{C\}$,$\{AB\}$,$\{AC\}$,$\{BC\}$,$\{ABC\}$上分别对

度量属性进行聚集操作后的并集。其中{}表示进行聚集运算{ * , * , * ,聚集函数 (*M*)}, {*A*}表示进行聚集运算{*A*, * , * ,聚集函数(*M*)}等。

这些聚集运算与操作结果是数据仓库中的一种高度综合级数据,实质上是进行了数据的浓缩(压缩),也可称为泛化。最终所获得的这些数据立方体可用于决策支持、知识发现,或其他许多应用。

例如,对表 3.12 所示的超市的基本数据集 POS(product,type,counter,price),前三个属性分别代表(产品名、类型、柜台)为维属性,对度量属性价格 price 进行取平均值(average)的聚集运算,则通过 Cube 操作可以得到一个具有三个维属性和一个度量属性的数据立方体 Dpos,如表 3.13 所示。也可以用三维方式来体现立方体的特征(省略)。

**表 3.12　基本数据集 POS**

| product | type | counter | price |
|---|---|---|---|
| KONKA | TV SET | 01 | 1000 |
| TCL | TV SET | 01 | 1500 |
| NOKIA | PHONE | 01 | 2000 |

**表 3.13　全聚集的数据立方体 Dpos**

| product | type | counter | M(AVG(price)) |
|---|---|---|---|
| * | * | * | 1500 |
| KONKA | * | * | 1000 |
| TCL | * | * | 1500 |
| NOKIA | * | * | 2000 |
| * | TV SET | * | 1250 |
| * | PHONE | * | 2000 |
| * | * | 01 | 1500 |
| KONKA | TV SET | * | 1000 |
| TCL | TV SET | * | 1500 |
| NOKIA | PHONE | * | 2000 |
| * | TV SET | 01 | 1250 |
| * | PHONE | 01 | 2000 |
| KONKA | * | 01 | 1000 |
| TCL | * | 01 | 1500 |
| NOKIA | * | 01 | 2000 |
| KONKA | TV SET | 01 | 1000 |
| TCL | TV SET | 01 | 1500 |
| NOKIA | PHONE | 01 | 2000 |

一般来说，在商业应用中，全聚集的数据占据的空间是原始数据空间的数百倍，另外它的更新维护也需要花费很长时间，所以计算聚集时应在聚集所占用的空间、CPU 处理时间和 OLAP 系统查询响应时间之间有一个权衡。故数据立方体的构建是在存储空间、响应查询时间和数据更新维护的消耗等几个主要因素之间寻求有效的折中，即部分物化：按照一定的规则选择数据立方体的一个子集进行预先计算。这种选择是存储空间和响应时间的一种折中。

典型的压缩型数据立方体，包括冰山立方体、紧凑数据立方体、外壳片段立方体等。随着流式数据处理技术的发展，流立方体生成方法越来越受到领域研究者的关注。

**2. 典型的压缩型数据立方体**

(1) 冰山立方体

在冰山立方体的生成计算中，仅聚集高于(或低于)某个阈值的子立方体，这是一种部分构建立方体的解决方法。这种计算方法的研究动机是数据立方体的空间多数被低(或高)度量值的数据单元所占据，而这些数据单元往往是分析者很少关心的内容。这种方法的优点是能够减少构建数据单元所占用的存储空间。

例如，在表 3.12 中，设定聚集运算条件：M(AVG(price))≤1250，其冰山立方体如表 3.14 所示。

**表 3.14 基本数据集 POS 的冰山立方体**

| product | type | counter | M(AVG(price)) |
|---|---|---|---|
| KONKA | * | * | 1000 |
| * | TV SET | * | 1250 |
| KONKA | TV SET | * | 1000 |
| * | TV SET | 01 | 1250 |
| KONKA | * | 01 | 1000 |
| KONKA | TV SET | 01 | 1000 |

对比表 3.14 和表 3.13，可以看出冰山立方体是全聚集立方体的部分。

(2) 紧凑数据立方体生成方法

紧凑数据立方体生成方法的一个重要特点是能够保持数据立方体的钻取操作的语义。这种紧凑数据立方体生成方法在压缩的方式和表现形式上表现出有不同的特征，其中包括浓缩立方体(Condensed cube)、商立方体(Quotient cube)等，这些都是近年来出现的一系列新型的数据立方体的存储结构。

浓缩立方体计算方法的基本原理是，在某些属性或组合下的一个元组相对于其他元组具有唯一性，则称为基本单一元组(BST，Base Single Tuple)。当它的超集(增加属性组合)也是 BST，且都是取同一度量值，在聚集运算时，可以把这些属性的度量值对应的元组压缩成一条元组存储。

例如，表 3.13 中的 product 属性下的每个元组都是基本单一元组 BST，由于其属性值

(KONKA,TCL,NOKIA)都是唯一的,同时,属性{ product }的所有超集{ product,type },{ product,counter },{ product,type,counter }也是 BST,且都具有相同值,如{ KONKA,*,*,1000 },{ KONKA,TV SET,*,1000 },{ KONKA,*,01,1000 },{ KONKA,TV SET,01,1000 },故可以将这些元组压缩存储为一条元组{ KONKA,*,*,1000 }。同理,属性{ type }中,其属性值为 PHONE 的元组是 BST,它和它的超集也可以压缩存储为一条元组{ *,PHONE,*,2000 }。经过这样的浓缩后,表 3.12 的基本数据集 POS 的浓缩立方体如表 3.15 所示。

**表 3.15 基本数据集 POS 的浓缩立方体**

| product | type | counter | M(AVG(price)) |
|---|---|---|---|
| * | * | * | 1500 |
| KONKA | * | * | 1000 |
| TCL | * | * | 1500 |
| NOKIA | * | * | 2000 |
| * | TV SET | * | 1250 |
| * | PHONE | * | 2000 |
| * | * | 01 | 1500 |
| * | TV SET | 01 | 1250 |

对比表 3.15 和表 3.13,可以看出浓缩立方体是全聚集立方体的有效浓缩。

由于在一般的应用中,当属性个数较多时,BST 是广泛存在的,一般来说,其压缩率可以达到 30%~70%。

(3) 外壳片段立方体

在高维情况下,需要预先计算大量的数据单元,同时增加了数据立方体的构建和维护复杂性。一种思路是仅预先计算涉及少数维度的子立方体,就形成整个数据立方体的一个外壳。当涉及其他维度的时候,则需要临时计算聚集结果。相关研究者提出仅计算其片段的方法,基于主要的观察是在 OLAP 过程中,只涉及少数的几个维度。

外壳片段立方体的计算方法的主要思想是:给定高维数据集,将维划分为互不相交的维片段,并且将每个片段转换成为倒排索引,然后构造外壳片段立方体。这样就可以利用预先计算的片段,动态组装和计算所需的子立方体单元。这种方法的优点是减少了计算数据立方体所需的数据空间,适用于高维数据的处理,同时能够快速响应涉及少量维度的查询。

(4) 流式数据立方体

现实世界的动态环境中产生的信息,构成了连续不断的流式数据。它分为事务型与度量型两种。

事务型流式数据:它产生于实体之间的交互日志,如大型综合网站的访问日志、金融交易信用记录等。

度量型流式数据:它来自于监控某个实体的状态,如传感器网络监控数据、气象观测数

据等、大型通信网络的异常检测、自然环境监测数据等。

用户从连续不断的流式数据中发现不同层次上的异常模式、兴趣模式、变化趋势等，为实时的在线决策提供强有力的支持。

流式数据立方体则是针对流式数据的多维分析提出来的解决方法。流式数据立方体模式表示为 SC=($T$,$D$,$M$)，其中 $T$ 为时间维度属性，$D$ 为非时间维度性集合，$M$ 为度量属性集合。由于流式数据量巨大，因此需要考虑部分物化策略减少存储空间消耗。一般通过兴趣视图子集选择、多层次时间窗口约束和适应性数据单元划分等策略限定流立方体所占用的存储空间。

### 3.4.5 多维数据分析的 MDX 语言及其应用

#### 1. MDX 语言简介

MDX(Multi Dimensional eXpressions，多维表达式)是联机分析处理(OLAP)和数据仓库应用中使用最广泛的软件语言(维度语言)。MDX 语言可以查询和管理多维数据仓库，MDX 表达式可以用来创建新的计算成员。

在语法的结构上，MDX 与 SQL 都包含“选择对象”(select 子句)、“数据源”(from 子句)以及“指定条件”(where 子句)，除这些关键字外，MDX 还结合了多维数据集，指定“维度”(On 子句)和“创建表达式计算的新成员”(MEMBER 子句)。这样就可用来从多维数据集中挖掘出指定的数据；MDX 语法还包含功能强大的函数，以协助数据处理与挖掘。具体说明如下：

(1) 关键字 SELECT 后带需要检索内容的子句。

(2) 关键字 ON 和维度(坐标轴)的名称一起使用，以指定数据库维度显示位置。

(3) MDX 用大括号{ }包含某个特定维度或者多个维度的一组元素。一个维度(度量维度或时间维度)的多个元素间用逗号(,)隔开。元素名称用方括号[ ]引用，并且不同组成部分之间用点号(.)分隔。

(4) 在一个 MDX 查询中，不同查询的维度(坐标轴)的数量可能不同。前三个坐标轴以 columns、rows 及 pages 命名，更多的坐标轴命名为 chapters、section 等。也可以统一用 axis(0)、axis(1)、axis(2)等表示坐标轴。

(5) MDX 查询中 FROM 子句指明用于查询数据的多维数据集。

(6) WHERE 子句指定在列或行(或者其他的坐标轴)上没有出现的多维数据集的成员。

#### 2. 多维数据查询

在多维数据集中用得最多的查询是对多维数据的切片查询，通过不同角度的切片来发现问题。下钻操作一般用来查询问题的原因。下面分别通过切片查询和向下钻取操作的例子进行说明。

**例 1**：切片查询。

在多维数据集 Sales 中，顾客所在的 MA 州，对时间 2009 年 Q1(1 季度)和 Q2(2 季度)的销售额 Dollar Sales 和销售数量 Unit Sales 的情况进行切片查询。

MDX 语言的切片查询语句：

```
SELECT
{ [Measures]. [Dollar Sales], [Measures]. [Unit Sales]}
On columns,
{ [Time]. [Q1, 2009], [Time]. [Q2, 2009] }
On rows
FROM [Sales]
WHERE ([Customer]. [MA])
```

切片查询结果见表 3.16。

**表 3.16　多维数据集 Sales 的切片查询**

| | Dollar Sales | Unit Sales |
|---|---|---|
| Q1,2009 | 96,949.1 | 3866 |
| Q2,2009 | 104,510.2 | 4125 |

**例 2**：向下钻取操作。

一种常用的查询是获得一个成员的子成员。这样做的目的是执行一个向下钻取操作，即获得基于一个共同父成员的范围内的成员。MDX 提供. Children 函数来完成这个操作。

下面将对多维数据集 Sales 中顾客所在的 TX 州进行向下钻取查询：工具产品[Product].[Tools]成员和它的子成员(Tool1,…,Tool5)，以及 2009 年 Q3(3 季度)成员的子成员(7、8、9)三个月的销售数量[Measures].[Unit Sales]情况。

MDX 语言的向下钻取操作语句如下：

```
SELECT
{ [Time]. [Q3, 2009]. Children } on columns,
{ [Product]. [Tools], [Product]. [Tools]. Children }
On rows
FROM Sales
WHERE ([Customer]. [TX], [Measures]. [Unit Sales])
```

Tools 成员及其子成员报表的各行上显示，如表 3.17 所示。

**表 3.17　使用. Children 的查询结果**

| | July,2009 | aug,2009 | sep,2009 |
|---|---|---|---|
| Tools | 176 | 266 | 205 |
| Tool1 | 32 | 121 | |
| Tool2 | | 78 | 85 |
| Tool3 | 57 | | 56 |
| Tool4 | 48 | 67 | |
| Tool5 | 39 | | 64 |

### 3. 创建新的计算成员

计算成员是通过对维成员进行设定的表达式计算后，所产生的新成员。创建新的计算成员需要引入关键字 WITH，称为"WITH 部分"。WITH 部分的位置在 SELECT 关键字之前。

在一个维度上定义一个计算成员的核心语法是：

```
MEMBER 成员标识符 AS '成员计算公式' [,properties…]
```

上面语法中的三个主要部分是：

- 成员标识符，它指定新建成员的名称以及定位该计算成员的维度和层次结构。
- 成员计算公式，它通过公式获得新计算成员的结果。
- 可选属性，提供额外的计算、显示以及其他的信息。

成员标识符必须包含在已知的维度名称中，作为其一个新的组成部分。成员计算公式中很可能涉及多个维度的成员计算公式，在各维度间如何控制计算公式的顺序呢？对每个计算成员都有一个相关的求解顺序（SOLVE ORDER），给定一个数字（整数 0、1 等），它表示该成员的计算优先级。

例如，在例 1 多维数据集中的 Dollar Sales 和 Unit Sales 两个成员的基础上，增加一个名为[Avg Sales Price]的新成员（进行除（/）运算）；在原有的 Q1，2009 和 Q2，2009 两个成员的基础上增加一个名为[Q1 to Q2 Growth]的新成员（进行减（—）运算）。对这两个新的计算成员的求解顺序给予规定，先算平均价格，后算增长量。

MDX 创建新计算成员语句为：

```
WITH
MEMBER [Measures]. [Avg Sales Price] AS
'[Measures]. [Dollar Sales]/[Measures]. [Unit Sales]',
SOLVE ORDER=0
MEMBER [Time]. [QI to Q2 Growth] AS
'[Time]. [Q2, 2009]-[Time]. [Q1, 2009]'
SOLVE ORDER=1
SELECT
{ [Measures]. [Dollar Sales], [Measures]. [Unit Sales],
[Measures]. [Avg Sales Price]}
on columns,
{ [Time]. [Q1, 2009], [Time]. [Q2, 2009], [Time]. [Q1 to Q2 Growth] }
On rows
FROM [Sales]
WHERE ([Customer]. [MA])
```

生成的新计算成员表见表 3.18。

表 3.18 利用除法和减法两公式生成的新计算成员表

| | Dollar Sales | Unit Sales | Avg Sales Price |
|---|---|---|---|
| Q1,2009 | 96,949.1 | 3866 | 25.08 |
| Q2,2009 | 104,510.2 | 4125 | 25.33 |
| Q1 to Q2 Growth | 7561.1 | 259 | 0.26 |

说明：在上面的 MDX 创建新计算成员语句中，若两计算公式颠倒一下顺序，其第 3 行第 3 列的元素的值就不是 0.26，而是 29.19 了。可见计算公式的顺序对计算结果是有影响的。

**4. MDX 语言更多功能**

MDX 语言中，允许的算术操作符有：加＋、减－、乘 * 、除/、括号( )。

允许的函数有：Avg()平均值；Aggregate()聚合函数定义的聚合值；Count()值或元组的个数；Sum()值的和；Max()最大值；Median()集的中位值；Min()最小值；Stdev()值的样本标准差；StdevP()值的总体标准差；Var()值的样本方差；VarP()值的总体方差等等。

允许额外的数字计算函数有：Abs(num)，num 的绝对值；Exp(N)，e 的 N 次幂；Factorial(N)，N 的阶乘；Ln(nun)，Num 的自然对数等等。

MDX 语言还具有更高级的功能，能帮助公司进行保险索赔分析、产品质量控制和顾客购物偏好等更高层次的决策分析。在此不多介绍，有兴趣的读者可参考有关资料。

## 习 题 3

1. 联机分析处理(OLAP)的简单定义是什么？它体现的特征是什么？
2. OLAP 准则中主要准则有哪些？
3. 什么是维？关系数据库是二维数据吗？如何理解多维数据？
4. MDDB 与 RDBMS 有什么不同？说明各自的特点。
5. 比较 ROLAP 与 MOLAP 在数据存储、技术及特点上的不同。
6. HOLAP 数据模型的特点是什么？
7. 举例说明多维数据显示的两种不同方法。
8. 举例说明多维类型结构(MTS)。
9. 举例说明四维数据显示。
10. 举例说明六维数据显示。
11. 多维数据显示的经验规则是什么？
12. 举例说明 OLAP 的多维数据分析的切片操作。
13. 举例说明 OLAP 的多维数据分析的钻取功能。
14. 说明四种不同的多维数据分析方法的作用。
15. 广义 OLAP 功能如何提高多维数据分析能力？
16. 说明数据立方体的概念。

17. 如何理解数据立方体就是数据仓库结构图中综合数据层？
18. 为什么要研究数据立方体的压缩技术？
19. 说明浓缩立方体的压缩方法和效果。
20. 多维数据分析的 MDX 语言与数据库的 SQL 语言有什么不同？
21. MDX 语言如何完成向下钻取操作？
22. MDX 语言如何完成表达式计算？

# 第4章 数据仓库设计与开发

## 4.1 数据仓库分析与设计

数据仓库分析与设计由需求分析、概念模型设计、逻辑模型设计与物理模型设计四个部分组成。

### 4.1.1 需求分析

数据仓库是一个向用户提供战略信息的环境，从而为用户提供决策支持。数据仓库不同于现行的事务处理系统(数据库应用系统)。事务处理系统完成每日的业务运行，用户所需的功能、信息内容、使用方式，系统有清楚的定义。数据仓库不能清楚地定义用户的需求，既不能准确定义用户真正想从数据仓库中得到哪些信息，也不能说明他们如何使用和处理这些信息。但是，用户可以说明哪些是重要的衡量指标，如何将各种信息综合起来为战略决策服务。

例如，市场部经理感兴趣的是每个月、某个地区、按照销售部门、参照历史数据和计划数据，了解新产品创造多少利润。销售经理需要按照产品种类，每天、每星期、每月进行汇总，按照销售地区或按销售渠道进行统计。财务经理在制定费用列表时，要与预算比较，按照每月、每季度和每年、按照预算资金定义、按照地区，对全公司进行汇总统计。

数据仓库的需求分析是数据仓库设计的基础。需求分析的任务是通过详细调查现实世界要处理的对象(企业、部门、用户等)，充分了解原系统(人工系统或计算机系统)工作概况，明确用户的各种需求(包括当前的需求和长远的需求)，为设计数据仓库服务。概括地说，需求分析要明确用哪些数据经过分析来实现用户的决策支持需求。

数据仓库用户包括高层主管、部门经理、IT专业人员等。通过对用户的调查，对数据仓库系统需要确定的问题如下。

(1) 确定主题域

① 明确对于决策分析最有价值的主题领域有哪些。

② 每个主题域的商业维度是哪些？每个维度的粒度层次有哪些？

③ 制定决策的商业分区是什么？

④ 不同地区需要哪些信息来制定决策？

⑤ 对哪个区域提供特定的商品和服务？

(2) 支持决策的数据来源

① 哪些源数据(数据库)与商品主题有关？

② 在已有报表和在线查询(OLTP)中得到什么样的信息？

③ 提供决策支持的细节程度是怎样的？

(3) 数据仓库的成功标准和关键性能指标

① 衡量数据仓库成功的标准是什么？

② 有哪些关键的性能指标？如何监控？

③ 对数据仓库的期望是什么？

④ 对数据仓库的预期用途有哪些？

⑤ 对计划中的数据仓库的考虑要点是什么？

（4）数据量与更新频率

① 数据仓库的总数据量有多少？

② 决策支持所需的数据更新频率是多少？时间间隔是多长？

③ 每种决策分析与不同时间的标准对比如何？

④ 数据仓库中的信息需求的时间界限是什么？

通过需求分析，明确为决策支持所需要的数据，包括如下内容：

（1）数据源

建立数据仓库需要使用源系统的数据，从这些源系统中收集、合并和整合数据，正确地转换这些数据，装入到数据仓库中。

数据源中的数据包括：

① 可用的数据源（数据库）；

② 数据源的数据结构；

③ 数据源的位置；

④ 数据源的计算机环境；

⑤ 数据抽取过程；

⑥ 可用的历史数据。

（2）数据转换

数据仓库中的数据是为决策分析服务，而源系统的数据为业务处理服务。这样需要决定如何正确地将这些源数据转换成适合数据仓库存储的数据。

在需求分析文档中要包括数据转换的细节，不但要明确从什么地方得到数据，还要描述在将数据载入数据仓库之前的合并、转化和分拆的过程。

（3）数据存储

通过对用户的采访，会发现数据仓库所需要的数据的详细程度，包括足够的关于存储需求的信息，估计数据仓库需要多少历史和存档数据。

（4）决策分析

需求分析文档应该包括用户决策分析的需求，即：

① 向下层钻取分析；

② 向上层钻取分析；

③ 横向钻取分析；

④ 切片分析；

⑤ 特别查询报表。

### 4.1.2 概念模型设计

将需求分析过程中得到的用户需求抽象为计算机表示的信息结构，即概念模型。它是

从客观世界(用户)到计算机世界的一个中间层次,即用户需求的数据模型。

概念模型的特点是:

(1) 能真实反映现实世界,能满足用户对数据的分析,达到决策支持的要求,它是现实世界的一个真实模型。

(2) 易于理解,有利于和用户交换意见,在用户的参与下,能有效地完成对数据仓库的成功设计。

(3) 易于更改,当用户需求发生变化时,容易对概念模型进行修改和扩充。

(4) 易于向数据仓库的数据模型(星型模型)转换。

概念模型最常用的表示方法是实体—关系法(E-R 法),这种方法用 E-R 图作为它的描述工具。E-R 图描述的是实体以及实体之间的联系,用长方形表示实体,在数据仓库中就表示主题,在框内写上主题名,椭圆形表示主题的属性,并用无向边把主题与其属性连接起来;用菱形表示主题之间的联系,菱形框内写上联系的名字,用无向边把菱形分别与有关的主题连接,在无向边旁标上联系的类型。若主题之间的联系也具有属性,则把属性和菱形也用无向边连接上。

由于 E-R 图具有良好的可操作性,形式简单,易于理解,便于与用户交流,对客观世界的描述能力也较强,在数据库设计方面更得到广泛的应用。因为目前的数据仓库一般建立在关系数据库的基础之上,与数据库的概念模型相一致,采用 E-R 图作为数据仓库的概念模型仍然是较为适合的。

通过一个例子来说明数据仓库的概念模型的设计,有两个主题:商品和客户,主题也是实体。

商品有如下属性组:

商品的固有信息(商品号、商品名、类别、价格等);

商品库存信息(商品号、库房号、库存量、日期等);

商品销售信息(商品号、客户号、售价、销售日期、销售量等);

其他信息等。

客户有如下属性组:

客户固有信息(客户号、客户名、性别、年龄、文化程度、住址、电话等);

客户购物信息(客户号、商品号、售价、购买日期、购买量等)。

其中商品的销售信息与用户的购物信息是一致的,它们是两个主题之间的联系。

将两个主题的概念模型用 E-R 图画出,如图 4.1 所示。

### 4.1.3 逻辑模型设计

逻辑模型设计是把概念模型设计好的 E-R 图转换成计算机所支持的数据模型。数据仓库在计算机中的数据模型是星型模型。这样,数据仓库的逻辑模型设计主要是将用 E-R 图表示的概念模型转换成星型模型。

数据仓库逻辑模型设计的主要工作为:

- 主题域进行概念模型(E-R 图)到逻辑模型(星型模型)的转换;
- 粒度层次划分;

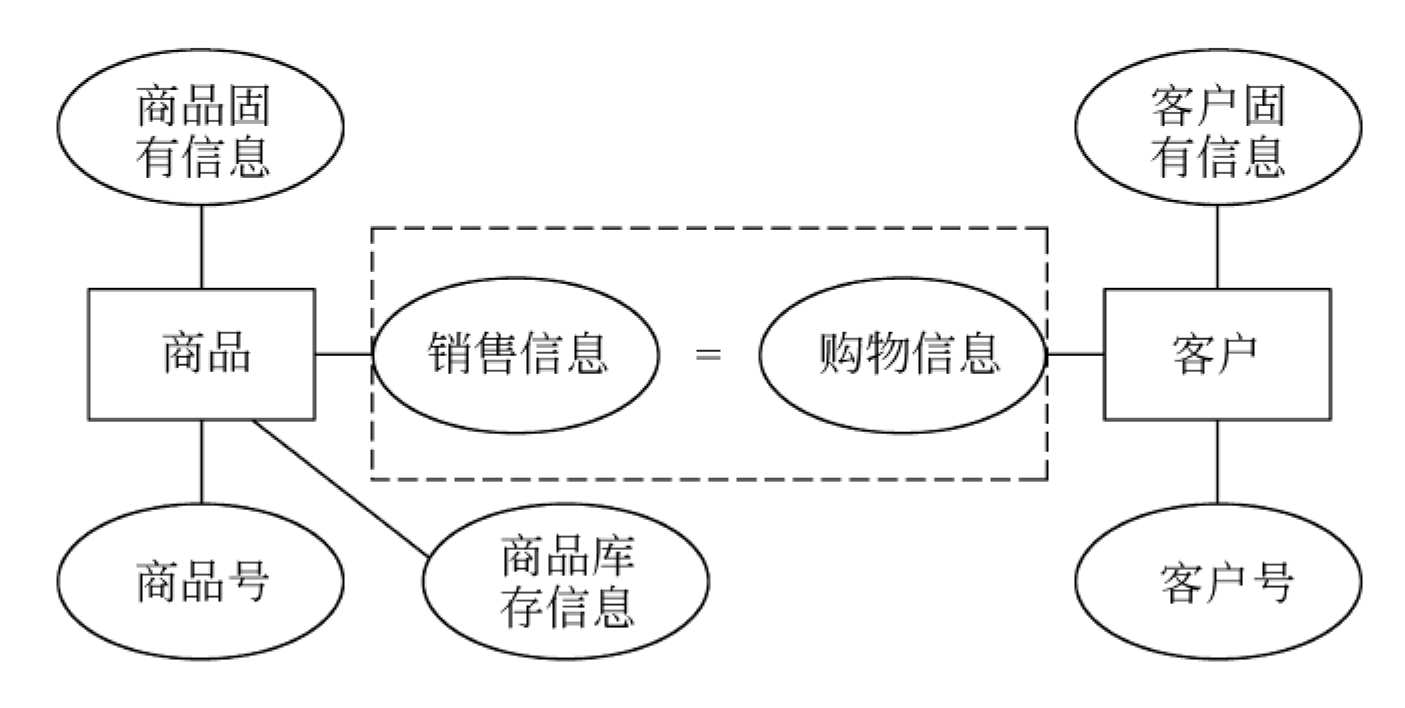



图 4.1　商品与客户两主题的概念模型

• 关系模式定义；
• 定义记录系统。

**1. 主题域进行概念模型到逻辑模型的转换**

在概念模型设计中,可能确定了多个主题域。但是,数据仓库的设计一般是从一个或几个主题逐步完成的。选择第一个主题域要足够大,使该主题能完成围绕该主题的决策分析需要。但又要足够精练,便于开发和较快实施。

例如,概念模型设计时,确定了“商品”和“客户”两个主题。其中“商品”对于商场来说是更基本的业务对象。商品的业务有销售、采购、库存等,其中商品销售是最主要的业务。它是进行决策分析最主要的方面。因而,“商品”主题比“客户”主题更重要。

星型模型的设计步骤如下:

(1) 确定决策分析需求

数据仓库是面向决策分析的,决策需求是建立多维数据模型的依据,例如分析销售额趋势、对比商品销售量、促销手段对销售的影响等。

(2) 从需求中识别出事实

在决策主题确定的情况下,选择或设计反映决策主题业务的表,例如在“商品”主题中,以“销售数据”作为事实表。

(3) 确定维

确定影响事实的各种因素,对销售业务的维一般包括商店、地区、部门、城市、时间、商品等,如图 4.2 所示。

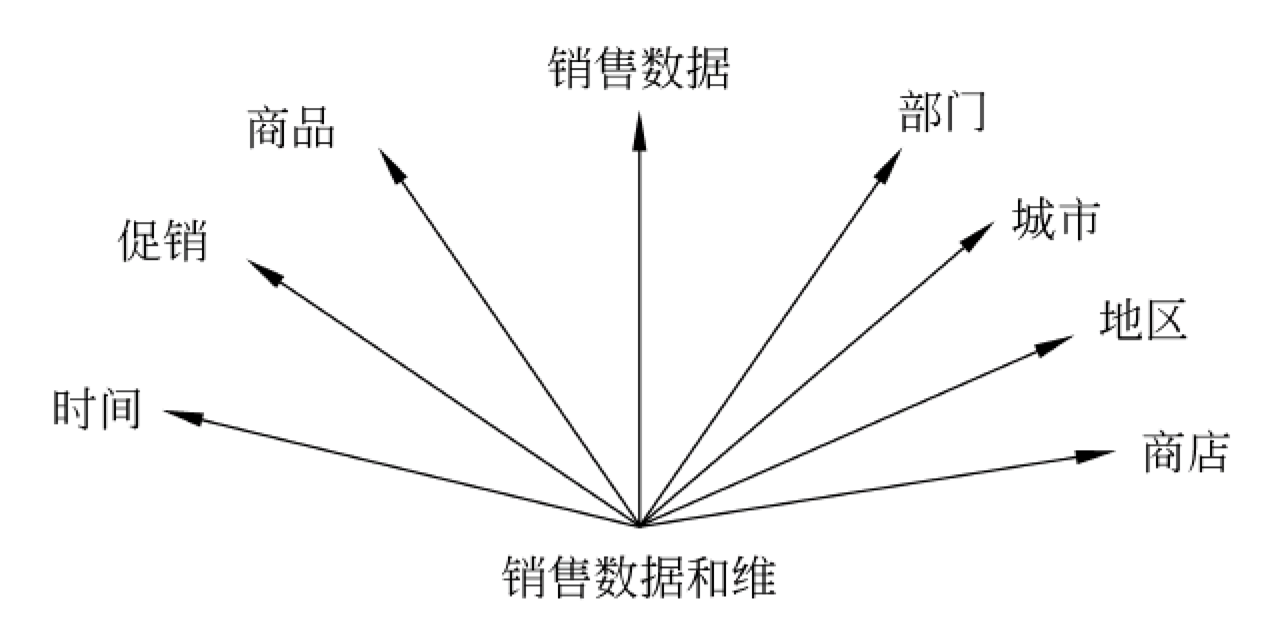



图 4.2　销售业务的多维数据

(4) 确定数据汇总的水平

存在于数据仓库中的数据包括汇总的数据。数据仓库中对数据不同粒度的综合,形成了多层次的数据结构。例如,对于时间维,可以用“年”、“月”或者“日”等不同水平进行汇总。

(5) 设计事实表和维表

设计事实表和维表的具体属性。在事实表中应该记录哪些属性是由维表的数量决定的。一般来说,与事实表相关的维表的数量应该适中,太少的维表会影响查询的质量,用户得不到需要的数据,太多的维表又会影响查询的速度。

(6) 按使用的 DBMS(数据库管理系统)和用户分析工具,证实设计方案的有效性

根据系统使用的 DBMS,确定事实表和维表的具体实现。由于不同的 DBMS 对数据存储有不同的要求,因此设计方案是否有效还要放在 DBMS 中进行检验。

(7) 随着需求变化修改设计方案

随着应用需求的变化,整个数据仓库的数据模式也可能会发生变化。因此在设计之初,充分考虑数据模型的可修改性可以降低系统维护的代价。

从概念模型的 E-R 图转换成逻辑模型的星型模型实例说明如下:

(1) 业务数据的 E-R 图

实体关系如图 4.3 所示。

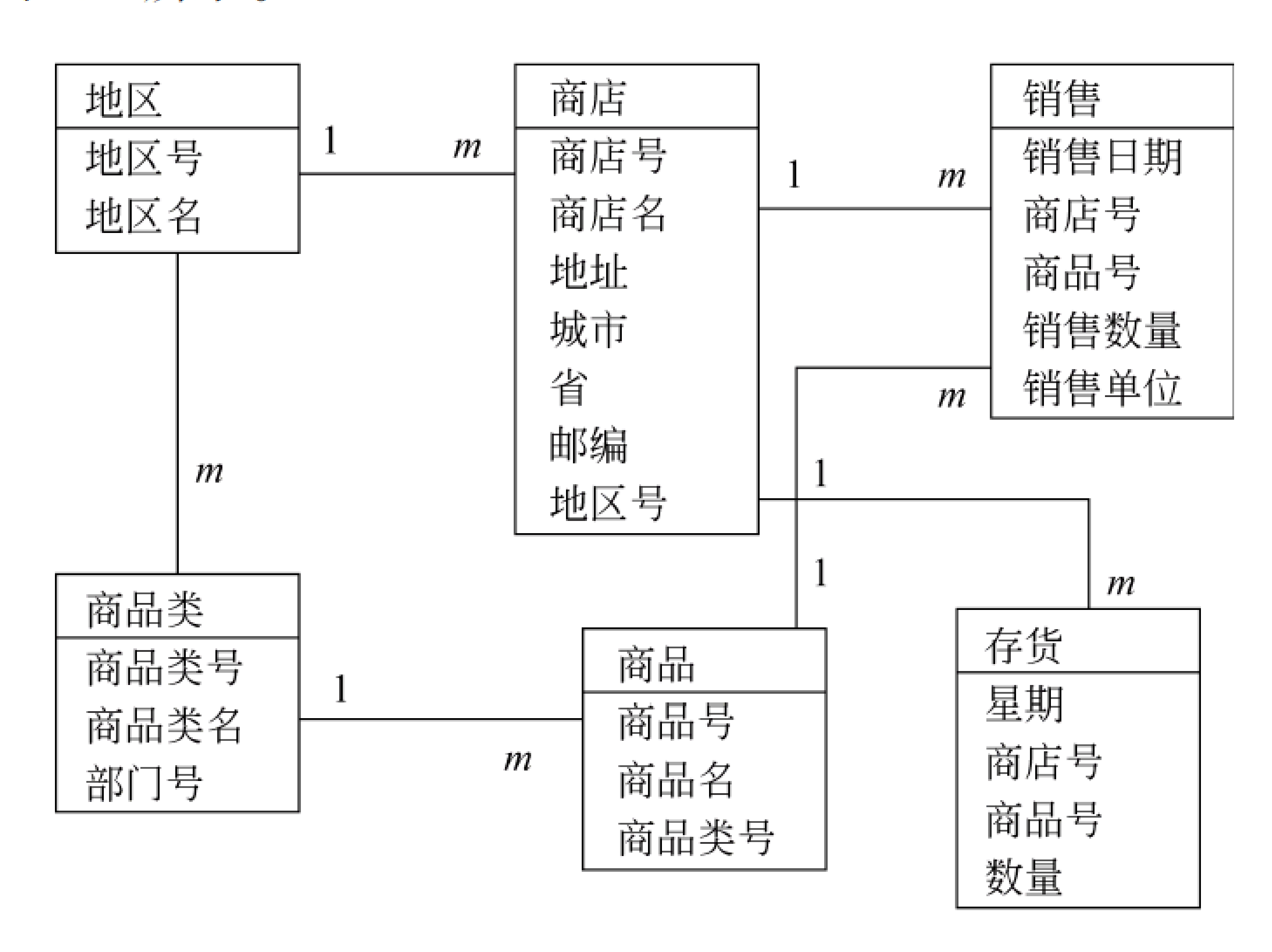



图 4.3 实体关系(E-R)图

(2) E-R 图向多维表的转换

该问题建立多维表模型时,先确定商品维和地区维。商品维包括部门、商品和商品大类,地区维包括地区和商店,忽略存货实体,建立销售事实。在 E-R 图中不出现的时间,在多维模型中增加时间维,如图 4.4 所示。

在多维模型中,实体与维之间建立映射关系,联系多个实体的实体就成为事实,此处销售实体作为事实,其他实体作为维,然后用维关键字将它转换为星型模型,如图 4.5 所示。

其中,地区维是综合了“地区”和“商店”两个实体,它们有一个层次的差别。将“商店”作为 1 级,“地区”作为 2 级,该维的关系表如表 4.1 所示。

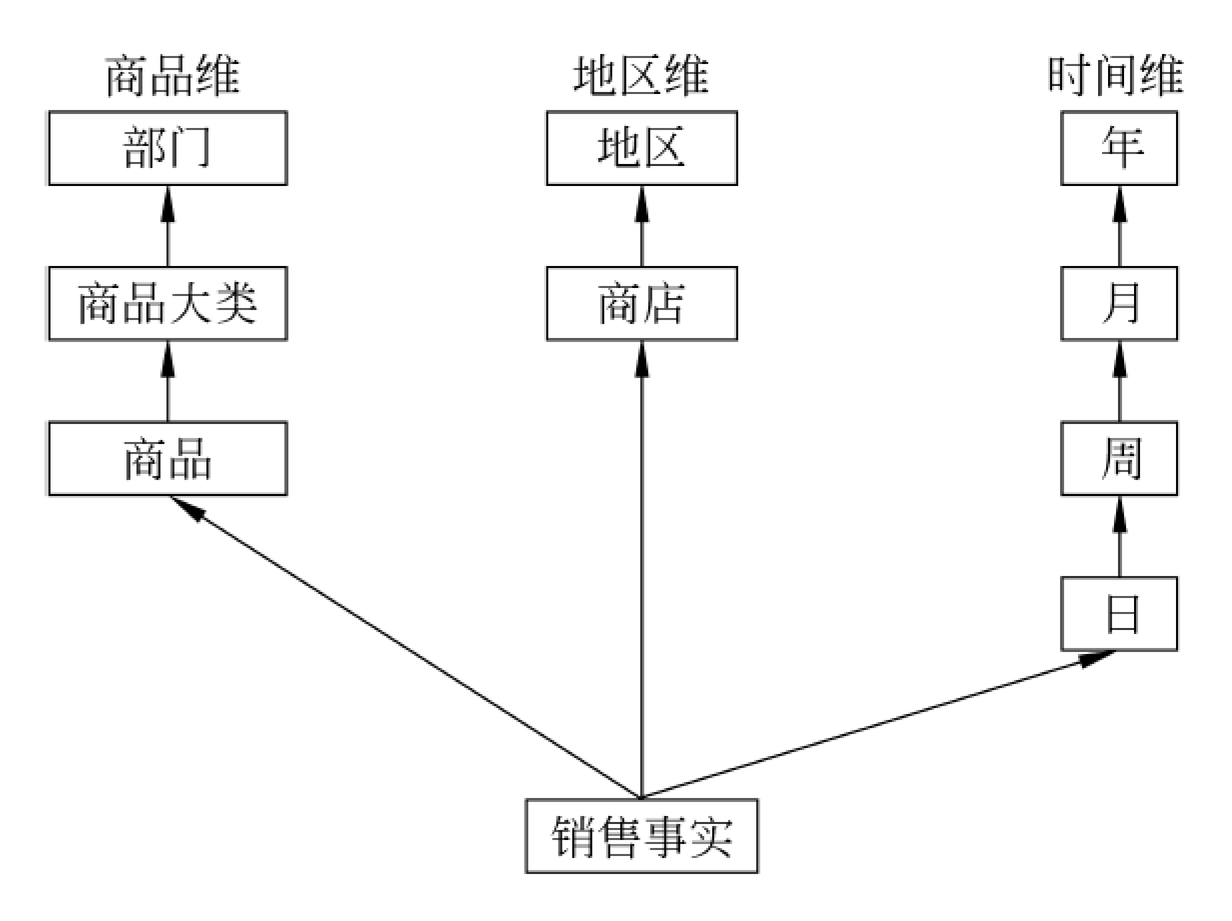



图 4.4　E-R 图向多维模型的转换

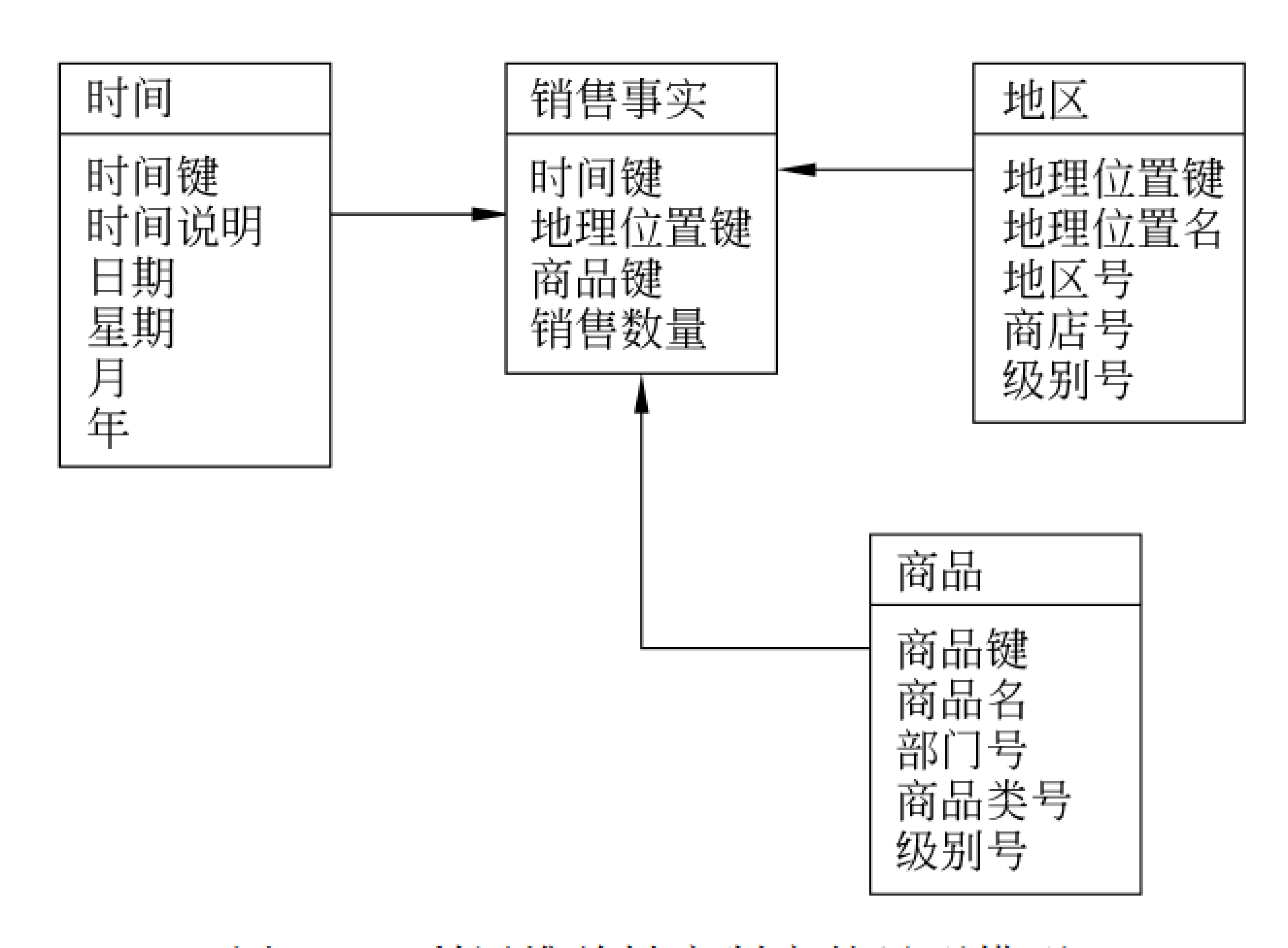



图 4.5　利用维关键字制定的星型模型

**表 4.1　地区维关系表**

| 地理位置键 | 地理位置名 | 地　区　号 | 商　店　号 | 级　别　号 |
|---|---|---|---|---|
| 100 | 东北地区 | 1 | | 2 |
| 105 | 中西部 | 2 | | 2 |
| 110 | 中南地区 | 3 | | 2 |
| 115 | 沈阳 | 1 | 2204 | 1 |
| 120 | 西安 | 2 | 2349 | 1 |
| 125 | 长春 | 1 | 2542 | 1 |
| 130 | 广州 | 3 | 2211 | 1 |

商品维综合了“商品”和“商品类”两个实体，它们也有一个层次的差别，同地区维一样处理。

在各维中，只有部门、商品类、地区、商店的编号没有具体的说明，为了打印报表将增加这些编号的名称说明，即部门名、商店名等，在维表中增加这些说明，即修改该星型模型如

图 4.6 所示。

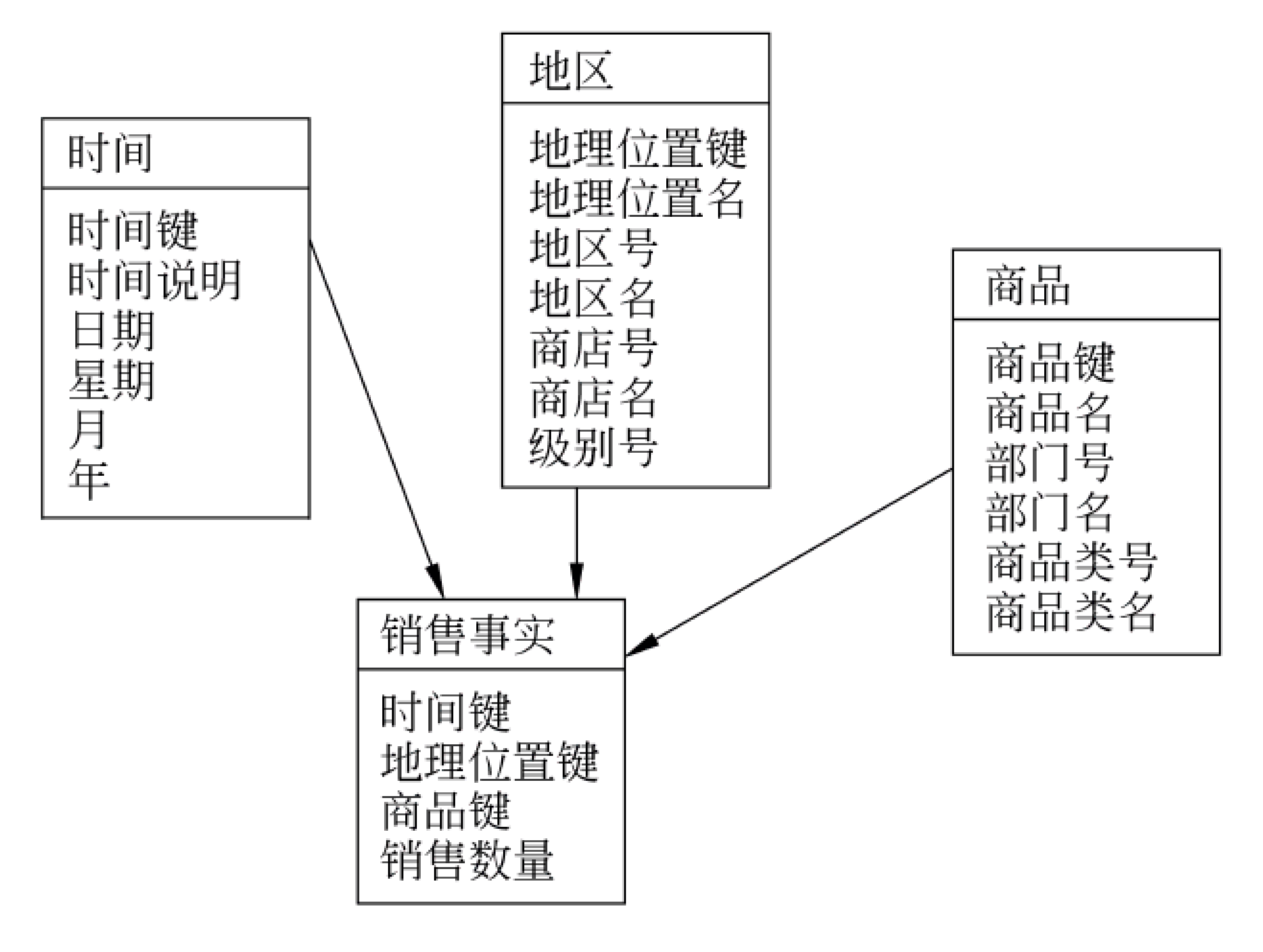



图 4.6　修改后的星型模型

**2. 粒度层次划分**

所谓粒度是指数据仓库中数据单元的详细程度和级别。数据越详细，粒度越小，层次级别就越低；数据综合度越高，粒度越大，层次级别就越高。在传统的事务处理系统中，对数据的处理和操作都是在详细数据级别上的，即最低级的粒度。但是在数据仓库环境中主要是分析型处理，粒度的划分将直接影响数据仓库中的数据量以及所适合的查询类型。一般需要将数据划分为详细数据、轻度综合、高度综合三级或更多级粒度。不同粒度级别的数据用于不同类型的分析处理。粒度的划分是数据仓库设计工作的一项重要内容，粒度划分是否适当是影响数据仓库性能的一个重要方面。

进行粒度划分，首先要确定所有在数据仓库中建立的表，然后估计每个表的大约行数。在这里只能估计一个上下限。需要明确的是，粒度划分的决定性因素并非总的数据量，而是总的行数。因为对数据的存取通常是通过存取索引来实现的，而索引是对应表的行来组织的，即在某一索引中每一行总有一个索引项，索引的大小只与表的总行数有关，而与表的数据量无关。

例如商场数据仓库的例子，一个商场可以经营上千种甚至更多的商品，商品的来源也有许多，每日的商品销售数据更是不计其数，每时每刻都在生成新的记录，进入“商品”主题的数据量是很大的，因而最好采用多重粒度，如对商品销售的分析主要是进行销售统计以及销售趋势分析，因此，定义商品销售数据的综合层次要更丰富一些，如每种商品（按商品号）的周统计销售数据、月统计销售数据以及季统计销售数据，每类商品（按商品类型）的周统计销售数据、月统计销售数据以及季统计销售数据等等。

**3. 关系模式定义**

数据仓库的数据最终将以关系数据库显示和存储。每个主题都是由多个表来实现的，这些表之间依靠主题的公共码键联系在一起，形成一个完整的主题。在进行概念模型设计

时，就确定了数据仓库的基本主题，并对每个主题的公共码键、基本内容等做了描述。在这一步里，将要对选定的当前实施的主题进行模式划分，形成多个表，并确定各个表的关系模式。

如对“商品”主题，考虑粒度划分层次，有如下关系表的内容。

公共码键：商品号。

（1）商品固有信息

商品表（商品号、商品名、类型、颜色、价格、……）——细节级

（2）商品销售信息

销售表 1（商品号、客户号、销售日期、售价、销售量、……）——细节级

销售表 2（商品号、时间段 1、销售总量、……）——综合级

……

销售表 $n$（商品号、时间段 $n$、销售总量、……）——综合级

### 4. 定义记录系统

数据仓库中的数据来源于多个已经存在的事务处理系统及外部系统。定义记录系统是建立数据仓库中的数据以源系统中的数据的对照记录。由于各个源系统的数据都是面向应用的，不能完整地描述企业中的主题域，并且多个数据源的数据存在着许多不一致，因此要从数据仓库的概念模型出发，结合主题的多个表的关系模式，需要确定现有系统的哪些数据能较好地适应数据仓库的需要。这就要求选择最完整、最及时、最准确、最接近外部实体源的数据作为记录系统，同时这些数据所在的表的关系模式最接近于构成主题的多个表的关系模式。记录系统的定义要记入数据仓库的元数据。

以商场的数据仓库为例，“商品”主题的有关内容分散在原有的销售子系统、库存子系统、采购子系统等事务处理的数据库中。不同数据源有关商品的信息有相交的部分，可能存在不一致的信息。从记录系统的要求出发，选择原有的分散数据库中最接近外部实体源的数据定义为数据仓库的记录系统。商品主题的记录系统在元数据中可描述如表 4.2 所示。

**表 4.2　记录系统的定义**

| 主题名 | 属性名 | 数据源系统 | 源表名 | 源属性名 |
|---|---|---|---|---|
| 商品 | 商品号 | 库存子系统 | 商品 | 商品号 |
| 商品 | 商品名 | 库存子系统 | 商品 | 商品名 |
| 商品 | 类别 | 库存子系统 | 商品 | 类别 |
| 商品 | 客户号 | 销售子系统 | 客户 | 客户号 |
| 商品 | 销售日期 | 销售子系统 | 销售 | 日期 |
| 商品 | 售价 | 销售子系统 | 销售 | 单价 |
| 商品 | 销售量 | 销售子系统 | 销售 | 数量 |
| 商品 | 库存量 | 库存子系统 | 库存 | 库存量 |
| 商品 | 仓库号 | 库存子系统 | 仓库 | 仓库号 |

说明：数据仓库中主题中的属性名要统一规范化。各源系统中的数据库中相关属性名，去掉不要的属性项，作为数据仓库和源系统的对比说明（记录系统的定义）放入元数据中。

### 4.1.4 物理模型设计

数据仓库的物理模型设计是为逻辑模型设计的数据模型确定一个最适合应用要求的物理结构（包括存储结构和存取方法）。

物理模型的设计所做的工作是估计存储容量，确定数据的存储结构，确定索引，确定数据存放位置，确定存储分配。它是数据存储的数据模型。

#### 1. 估计存储容量

物理模型重点在于物理存储，随着数据仓库的增大需要知道最初和后来需要多少存储空间。

(1) 对每一个数据库表确定数据量

① 行（记录行）数的初始估计；

② 行的平均长度；

③ 估计行的每月增长数；

④ 表的初始大小，以兆字节（MB）计算；

⑤ 表按时间 6 个月和 12 个月存储的数据大小。

(2) 对所有的表确定索引

① 索引的个数；

② 索引对最初、6 个月和 12 个月存储数据所需要的空间。

(3) 估计临时存储

① 排序、合并需要的临时空间；

② 准备区（大量数据交换的场所）内的临时文件；

③ 准备区内的永久文件。

#### 2. 确定数据的存储计划

确定数据的存储计划包括以下内容。

(1) 建立汇总（聚集）计划

假设数据仓库用户有 80％的查询需要汇总信息，这样就应该建立汇总表。如果数据仓库只存储最小粒度的数据，每次查询遍历所有的明细记录，然后生成汇总信息，就要用去大量的时间。汇总（聚集）数据表必须包括在物理模型中。应该建立多少汇总表，这要根据查询需求来决定。

(2) 确定数据分区方案

假设有 4 个维表，平均每个表有 50 行，对于这些维度表中的行，潜在的事实表将有超过 600 万行记录。事实表非常巨大，大表非常难以管理。

分区可以将表分解成易于管理的小表。对事实表的分区并不是简单地分解数量。一般

采用按垂直分区或水平分区(即按不同维度分区或按时间顺序分区),制定分区准则(如按产品分组)。除事实表分区外,维表也分区。每个表的分区个数是多少,在表分区后,使查询知道到所需的分区内进行。

(3) 建立聚类选项

在数据仓库中,很多的数据访问是基于对大量数据的顺序访问,这可以通过聚类来提高性能。聚类是将相关的数据放在存储介质的相邻物理块上进行管理。这种安排使相关联的数据能够在一次输入操作中全部取出,提高查询效率。

### 3. 确定索引策略

在数据仓库中由于数据量很大,需要对数据的存取路径进行仔细设计和选择,建立专用的复杂的索引,以获得最高的存取效率,因为在数据仓库中的数据是不常更新的,也就是说每个数据存储都是稳定的。虽然建立索引有一定的代价,但是一旦建立就几乎不需要再维护索引。

传统的数据库采用 B-Tree 索引,它是一个高效的索引,如图 4.7 所示。B 树是一个平衡(balance)树,即每个叶结点到根结点的路径长度相同。B 树索引是一个多级索引。每个非叶结点包括多个按顺序排列的关键字值:

$$K_1 < K_2 < \cdots < K_{n-1}$$

每个关键字有一个对应的指针 $P_i(i=1,2,\cdots,n-1)$指向下层结点的指针桶(多个关键字和对应的指针)最小关键字值。叶结点的关键字值的指针指向一个文件记录。

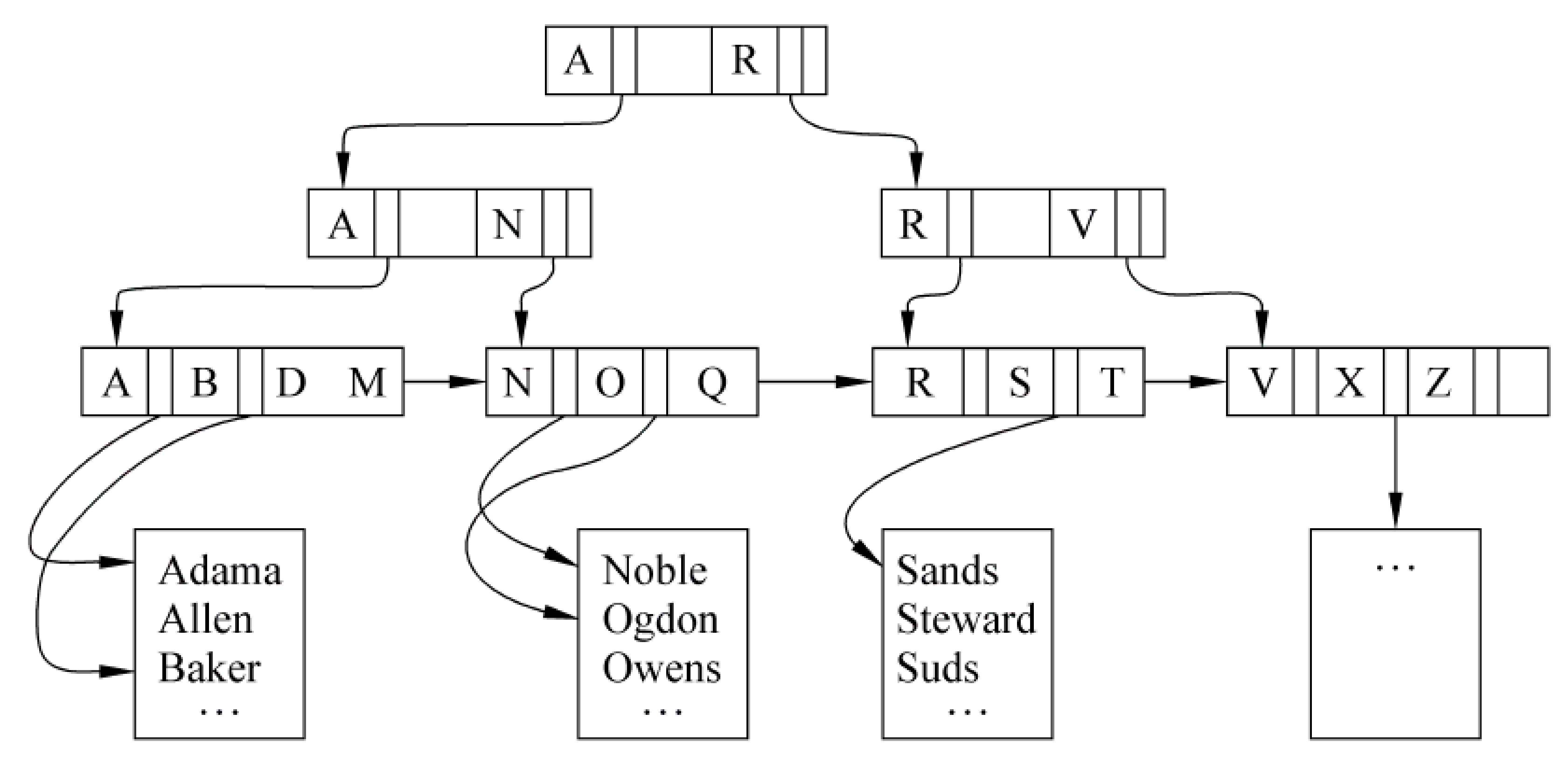



图 4.7　传统 B-Tree 索引

### 4. 确定数据存放位置

数据仓库中,同一个主题的数据并不要求存放在相同的介质上。在进行物理设计时,常常要按数据的重要程度、使用频率以及对响应时间的要求进行分类,并将不同类的数据分别存储在不同的存储设备中。重要程度高、经常存取并对响应时间要求高的数据就存放在高速存储设备上,如硬盘;存取频率低或对存取响应时间要求低的数据则可以放在低速存储设备上,如磁盘或磁带。

数据存放位置的确定还要考虑到的一些其他方法,如决定是否进行合并表;是否对一些

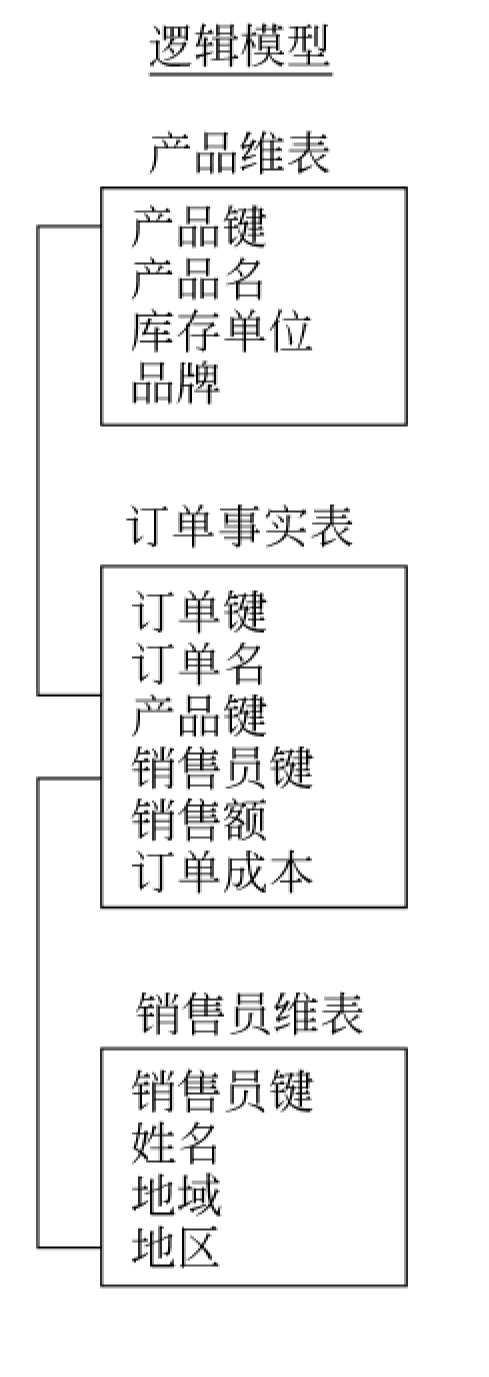
经常性的应用建立数据序列;对常用的、不常修改的表或属性是否允许冗余存储。如果采用了这些技术,就要记入元数据。

### 5. 确定存储分配

物理存储中以文件、块和记录来实现。一个文件包括很多块,每个块包括若干条记录。文件中的块是数据库的数据和内存之间 I/O 传输的基本单位,在那里对数据进行操作。

增大文件中的块大小,可以将更多的记录和行放入一个块中,因为一次读操作可以读入更多的记录,大块减少了读操作的次数。但是,大块结构对读取记录少时,操作系统也将读入很多不必要的信息到内存中,影响了内存管理。

用一个简例来说明逻辑模型和物理模型的内容,如图 4.8 所示。




物理模型

| 名称 | 类型 | 长度 | 注释 |
|---|---|---|---|
| 产品维表 | | | 包括公司所有产品的信息 |
| Product-Key | integer | 10 | 主键 |
| Product-Name | char | 25 | 产品名称 |
| Product-SKu | char | 20 | 库存单位 |
| 销售员维表 | | | 包括不同地区的所有销售员信息 |
| Salpers-Key | integer | 15 | 主键 |
| Salpers-Name | char | 30 | 销售员姓名 |
| Territory | char | 20 | 销售员所在区域 |
| Region | char | 20 | 所在地区 |
| 订单事实表 | | | 包括公司收到的所有订单 |
| Order-Key | integer | 10 | 订单键 |
| Order-Name | char | 20 | 订单名称 |
| Product-ref | integer | 10 | 参考产品主键 |
| Salpers-ref | integer | 15 | 参考销售员主键 |
| Order-Amount | Num | 8, 2 | 销售额 |
| Order-Cost | Num | 8, 2 | 订单成本 |

图 4.8　逻辑模型与物理模型

## 4.1.5　数据仓库的索引技术

索引技术的作用在于提高数据仓库访问效率。下面介绍三种重要的数据仓库索引技术:位索引技术、标识技术与广义索引。

### 1. 位索引技术

Sybase 公司推出的数据仓库 Sybase IQ,采用位索引(Bit-Wise)技术,它在处理复杂的查询时,比传统数据库索引 B-Tree 有了突破。

(1) Bit-Wise 索引技术

Bit-Wise 索引技术在存储数据的方式上与传统的关系数据库有所不同,它不是以"行记录"而是以"列"为单位存储数据,即对数据进行垂直分割。对于每一个记录的字段满足查询

条件的真假值用“1”或“0”的方式表示，或者用该字段中不同取值（即多位二进制）来表示。

一般辅助决策的查询往往仅涉及大量数据记录中的少数列，因而不需要访问原始数据就能快速获得查询结果。显然，利用字段的不同取值也能快速进行数据聚类、分组、求最大值、最小值及平均值等。

对于高度可选择的数（称高基数），如姓名或地址等可能有数万个选择值，用(1,0)真假值来索引是不合适的。

例如，检索“美国加州有多少男性未申请保险?”

在数据库中，每个记录中对于性别是男性的字段取值为1，女性为0，是加州的字段取值为1，其他为0，对于未参加保险的字段取值为1，参加的取值为0。该三列字段值为1或0。对三字段均满足条件记录进行累加。对下面的简单数据库利用Bit-Wise技术得到有两个记录满足条件，如图4.9所示。

| | 性别 | 保险 | 州 |
|---|---|---|---|
| 1 | M | Y | MA |
| 2 | M | N | CA |
| 3 | F | Y | IL |
| 4 | M | N | CA |

| 男 | 未保险 | 加州 | |
|---|---|---|---|
| 1 | 0 | 0 | |
| 1 | 1 | 1 | =2 |
| 0 | 0 | 0 | |
| 1 | 1 | 1 | |

图4.9　Bit-Wise索引

(2) B-Tree技术与Bit-Wise索引技术对比

Bit-Wise索引技术比B-Tree技术能提高响应速度10～100倍。

① B-Tree索引技术特点

- 按行存储数据；
- 针对具体查询来建立查询驱动的索引机制；
- 存储被索引的字段数据；
- 一列允许一个索引；
- 适合于高基数字段。

② Bit-Wise索引技术特点

- 按列存储数据；
- 针对实际特征建索引；
- 不存储实际索引字段内容；
- 一列允许多个索引；
- 数据压缩技术和位操作技术；
- 适合于低基数字段，兼顾高基数字段。

③ 实例比较

以检索“美国加州有多少男性未申请保险?”为例，假设数据库有10M记录，每个记录长800字节，每一页16K字节。

- 按传统的关系数据库的检索

需要经过50万次I/O操作。

- 按Bit-Wise检索

对于10M个记录建立三列的Bit-Wise索引，共占(10Mbit×3列/8)字节的空间，每页16K字节，则这些索引仅占235页。存取这些索引只需进行235次I/O操作。

④ B-Tree不适合数据仓库

- B-Tree只适合于高基数(Cardinality)字段

对于高基数字段，如物资编号、顾客编号等具有唯一的数据值，B-Tree 很适合。但对于低基数字段就毫无价值，如性别字段，只有男、女两个值，建立 B-Tree 索引就没有意义。

- B-Tree 索引增加了在数据仓库中构造和维护索引的代价

由于 B-Tree 索引包含实际数据和其他信息(如指针等)，因而使得索引需占用一定的空间和时间。如果构造所有相关的索引，数据仓库就会占 2～4 倍原始数据空间。当成批插入删除时，索引就非常敏感，有可能失去平衡并降低性能。通常来说，10%～15%的数据修改会导致重建索引。

- B-Tree 索引不适合复杂查询

B-Tree 用于简单查询及已知公共存取路径的环境下才有优点，而在数据仓库应用中，通常是复杂的查询，并经常带有分组及聚合条件。此时，B-Tree 索引往往无能为力。

**2. 标识技术**

使用标准的数据库技术来储存数据仓库是非常昂贵的。较好的替代方法是用基于标识的技术来储存数据仓库。这种技术根本不同于关系数据库技术。利用关系数据库技术，当加入一个记录到系统中时，会追加此数据的一个物理代表块到磁盘上。假设一些标准数据库管理系统中的样本记录如下：

| | 姓名 | 籍贯 | 职称 | 年龄 |
|---|---|---|---|---|
| 记录 1 | 陈文东 | 江西 | 教授 | 56 |
| 记录 2 | 何玉辉 | 河北 | 讲师 | 32 |
| 记录 3 | 李宝 | 湖南 | 副教授 | 37 |
| 记录 4 | 施东 | 江苏 | 讲师 | 28 |
| 记录 5 | 曹文杰 | 湖南 | 副教授 | 36 |
| 记录 6 | 赵玉 | 吉林 | 讲师 | 32 |
| 记录 7 | 黄小斌 | 江苏 | 讲师 | 28 |
| 记录 8 | 赛英花 | 山东 | 副教授 | 32 |
| 记录 9 | 彭宏 | 江西 | 讲师 | 25 |
| 记录 10 | 廖宇宙 | 湖南 | 教授 | 42 |

每次完成一个事务时，就会添加一个新记录到标准的数据库中。数据的缩放比例是线性的，因为数据量是存放多少记录的一个函数。但是在如上面所示的小型的、简单的数据库查看数据记录，会发现在整个数据库中有数据冗余。例如籍贯“湖南”出现了三次，年龄“32”则出现了 3 次，职称“讲师”出现了 5 次。因此这个数据库中有明显的物理冗余。

假设可以为此数据库中的每个实体创建一个标识。“江西”在籍贯中是 01 标识。“28”在年龄中是 02 标识。“讲师”在职称名中有一个 03 标识。上面的数据库可以被简化为一系列标识：

| 姓名 | | 籍贯 | | 职称 | | 年龄 | |
|---|---|---|---|---|---|---|---|
| 陈文东 | 01 | 江西 | 01 | 教授 | 01 | 25 | 01 |
| 何玉辉 | 02 | 河北 | 02 | 副教授 | 02 | 28 | 02 |
| 李宝 | 03 | 湖南 | 03 | 讲师 | 03 | 32 | 03 |

| | | |
|---|---|---|
| 施东 04 | 江苏 04 | 36 04 |
| 曹文杰 05 | 吉林 05 | 37 05 |
| 赵玉 06 | 山东 06 | 42 06 |
| 黄小斌 07 | | 56 07 |
| 赛英花 08 | | |
| 彭宏 09 | | |
| 廖宇宙 10 | | |

一旦建立完这些标识，数据库可被精简如下：

记录 1 01,01,01,07
记录 2 02,02,03,03
记录 3 03,03,02,05
记录 4 04,04,03,02
记录 5 05,03,02,05
记录 6 06,05,03,03
记录 7 07,04,03,02
记录 8 08,06,02,03
记录 9 09,01,03,01
记录 10 10,03,01,06

记录被标识以后，存储这些记录的空间将大大缩小。此外，数据量越大（也就是记录量越多），标准的数据库和标识数据库的存储需求差异也就越大。换句话说，记录量越多，基于标识的数据库的优势就越明显。使用标识数据库技术时，有几项非常有利的应用：

- 大量压缩数据。
- 数据越多，标识数据比标准的、基于记录的数据更有利。
- 因为数据被大量压缩，所以整个数据库可以存放在内存中。
- 可以索引所有的行和所有的列。

一旦将基于标识的数据库存放在内存中，处理速度会得到很大的提高。根据不同的细节，查询的速度可以提高两到三个（甚至更多）数量级。提高了处理速度，很多工作就会成为现实。例如，分析员可以很容易地进行扫描整个数据库的查询。

大量压缩数据的另一个主要益处就是索引所有属性成为可能。一旦可以索引所有属性，对数据仓库的探索分析就没有限制。分析员可以用任何需要的方式查看任意字段。查询的速度就像这样：如果分析员要精练结果，可以重新书写一个查询公式并重新运行。所有的这些重写公式表示和重新计算都可以在很短的时间里完成，这个时间远远少于标准的基于记录的数据库所需要的时间。事实上，探索数据仓库的功效依赖于基于标识的数据库技术。

### 3. 广义索引

对数据仓库的一个很广泛的应用问题是“这个月销售最好和最差的 10 种商品是哪些?”，可以设计这么一块“黑板”，在上面标明当月销售最好和最差的 10 种商品的名称或者

它们相关记录的存放地址。这块“黑板”就是通常所说的“广义索引”。

数据仓库的数据量巨大，所以要依靠各种各样的索引技术来提高涉及大数据量的查询的速度。“广义索引”在处理如上的最值(最大值或最小值)问题时，效果是非常明显的，也较易于实现。在从数据库环境抽取数据并向数据仓库中装载的同时，就可以根据用户的需要建立许多这样的“广义索引”。每次数据仓库装载时，就重新生成这些“广义索引”的内容。这样就并不需要为了建立“广义索引”而去扫描数据仓库。而且这些索引都非常小，开销也相当小，但它给应用所带来的便利却是显而易见的。对于一些经常性的查询，利用一个规模小得多的“广义索引”总要比去搜索一个大得多的关系表方便得多。

但是，同时出现的问题就是随着数据仓库“年龄”的增长以及数据仓库随时间变化的特性，这种“广义索引”的数目也就会成倍地增长，管理这些数目多、规模小、名目繁多的“广义索引”也就成为一件非常棘手的事情。这就需要在元数据中完整地定义说明这些“广义索引”。应用需要时，首先去查找元数据再去找相应的“广义索引”或表。

# 4.2　数据仓库开发

## 4.2.1　数据仓库开发过程

数据仓库的开发主要是围绕数据仓库功能展开的，数据仓库的主要功能包括数据获取，数据存储和决策分析，这三个功能模块组成了数据仓库的体系结构。随着决策需求的扩大，数据仓库的数据将迅速增长。数据仓库的开发要适应这种变化，采用螺旋式周期性的开发方法比较合适。

数据仓库的开发过程分为 4 个阶段 12 个具体步骤，如图 4.10 所示。

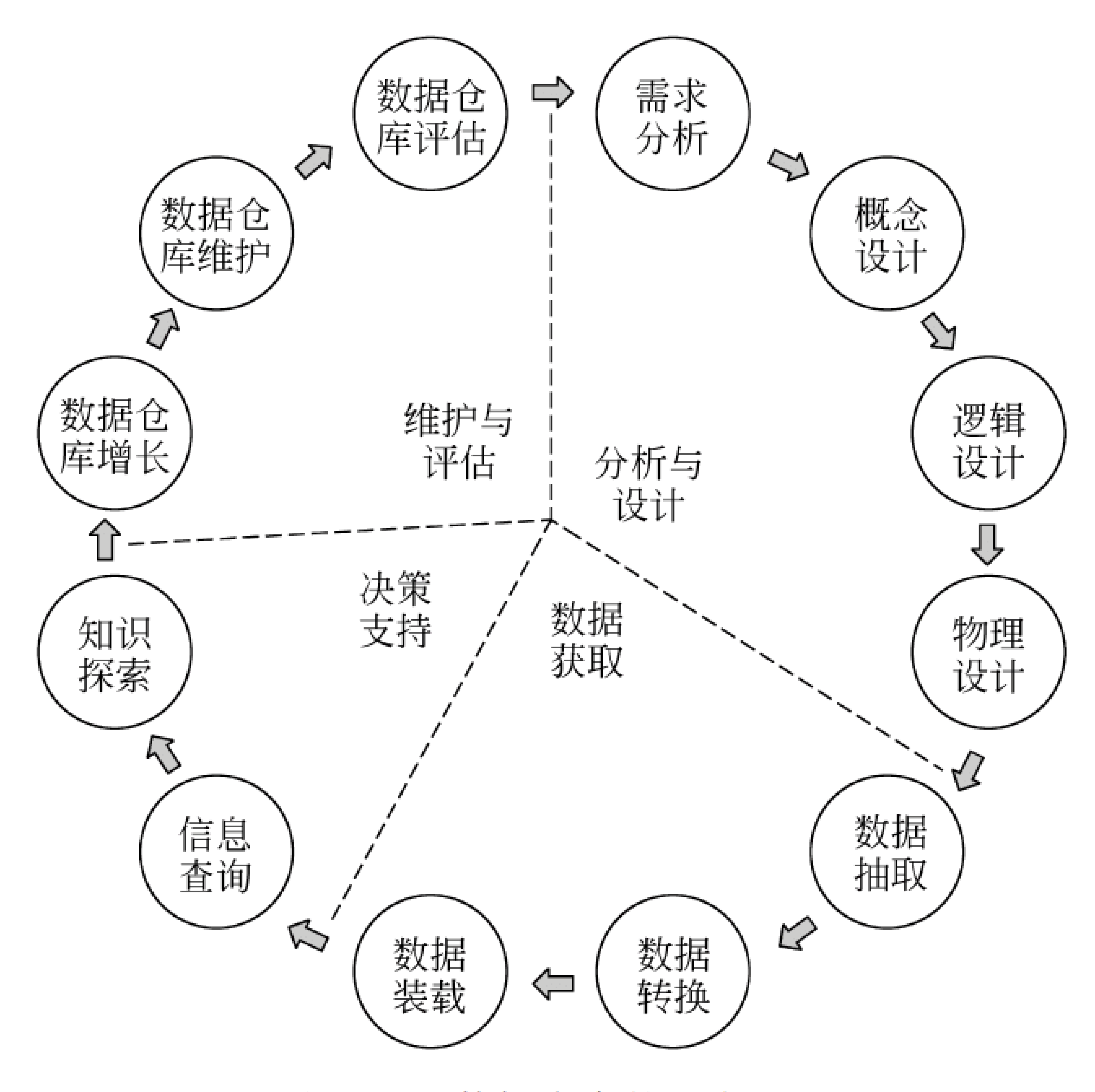



图 4.10　数据仓库的开发过程

#### 4.2.1.1 分析与设计阶段

数据仓库开发需要明确如下问题：

- 数据仓库开发的范围多大？这包括数据的范围、技术的作用(要用到新技术吗?)以及时间上的考虑(开发工作需要在多长时间内完成?)。
- 企业业务方面的驱动因素是什么，要解决的业务问题是什么？
- 开发的数据仓库的决策支持能力是什么？

数据仓库开发的分析和设计阶段包括需求分析、概念设计、逻辑设计和物理设计 4 个步骤。

**1. 需求分析**

数据仓库的需求分析是根据用户的决策支持需求，确定决策主题域，并分析主题域的商业维度，同时分析支持决策的数据来源，以及向决策主题数据的转换；整个数据仓库的数据量大小以及数据更新的频率确定决策分析方法等。

需求分析是设计和实现数据仓库的基础。

例如，银行业数据仓库的需求分析包括：

(1) 决策支持需求：在竞争性的市场中银行决策者认识到，它必须利用其日常活动中包含的大量信息，预测信用卡使用状况和利润率的能力。

(2) 信息需求：对最终用户进行调查以确定哪些信息有助于销售或有助于调整银行的信息政策。

(3) 业务需求：定义销售信息处理、信息的类型和销售渠道。

(4) 用户访问需求：确定用户访问数据仓库所需的时间，以及数据访问的偏好。

(5) 选择主题：选择一个主题区——“信用卡”。

(6) 初始规模：确定主题域的数据量。

**2. 概念设计**

在数据仓库的概念模型设计中，需要确定主题域及其内容。利用需求分析的结果建立概念模型，即对每个决策主题与属性以及主题之间的关系用 E-R 图模型表示出来。E-R 图能有效地将现实世界表示成信息世界，它也有利于向计算机的表示形式进行转化。

例如，银行业信用卡主题域分析。

(1) 主题域范围：确定了“信用卡”主题域，对某些实体，如顾客，要求它在这一主题域发挥作用。

(2) 所需细节水平：为支持概括和趋势计算，需要存入持卡人的日常活动。

(3) 初步概括表：对“信用卡”主题需要建立初步概括表，按行业和地理特征进行概括，将概括时段确定为每月。

**3. 逻辑设计**

在逻辑模型设计中，需要分析主题域，将概念模型(E-R 图)转换成逻辑模型，即计算机

表示的数据模型。数据仓库的数据模型一般采用星型模型。

逻辑设计中还需要进行数据粒度层次的划分;星型模型中事实表、维表的关系模式定义;数据转换的记录系统的定义。

银行业信用卡主题的逻辑模型是多维表的星型模型,需要将概念模型的 E-R 图转换成星型模型。

#### 4. 物理设计

数据仓库的物理模型设计是对逻辑模型设计的数据模型确定物理存储结构和存取方法。数据仓库的星型模型在计算机中仍用关系型数据库存储。

物理设计还需要进行存储容量的估计;确定数据存储的计划;确定索引策略;确定数据存放位置以及确定存储分配。

例如,银行业的物理数据库设计包括:

(1) 数据库设计:对主题中的事实表和维表设计数据库存储结构和存放位置。

(2) 概括表:按行业代码或按月建立一个概括表。

(3) 索引:对数据仓库中的数据建立多种索引。

(4) 建立备份和恢复准则:使数据仓库能适应不同的备份和恢复。为了防止数据损失,需要对文件进行备份。

### 4.2.1.2 数据获取阶段

它包括数据抽取、数据转换、数据装载 3 个步骤。

数据仓库中的数据主要来源于事务处理系统中的数据。由于数据仓库对数据的使用目的与事务处理对数据的使用的目的不同,这就形成了对事务处理系统中的数据的抽取,并进行转换,按数据仓库的数据存储要求装载数据。

#### 1. 数据抽取

数据抽取工作主要进行数据源的确认,确定数据抽取技术,确认数据抽取频率,按照时间要求抽取数据。

源系统的差异性,如计算机平台、操作系统、数据库管理系统、网络协议等的不同造成了抽取数据的困难。

#### 2. 数据转换

数据抽取得到的数据是不能直接存入数据仓库的。数据转换工作包括:数据格式的修正、字段的解码、单个字段的分离、信息的合并、变量单位的转化、时间的转化、数据汇总等。

#### 3. 数据装载

经过数据转换的数据装入数据仓库有三种类型:

- 初始装载:第一次装入数据仓库。
- 增量装载:根据定期应用需求装入数据仓库。

• 完全刷新：完全删除现有数据，重新装入新的数据。

数据装载时，一般利用选定的批量装载程序，目的是高效和及时地把数据装载到数据仓库中去。

例如，银行业的数据仓库的数据获取阶段包括：

(1) 候选数据源：给定数据需求和粒度需求，指定日常事务文件为关键数据源。

(2) 完整性：检查数据来源的完整性。

(3) 评价：对数据源进行评价。

(4) 数据转换：将数据源中的数据变换到目的地去，同时保持数据准确性和完整性的过程。

(5) 数据装载：将数据转换后的数据加载到目的文件和平台上去。可以用查询来验证业务报表的内容。

(6) 评审过程：开发评审程序来验证是否所有的信用卡事务都发生在指定的时间期限中。

(7) 元数据的加载：加载一般的元数据外，还要加载有特别用途的元数据，如在特殊环境中，反映数据变化的元数据。

(8) 系统测试：系统测试用以保证各部分能相互配合，并维护数据的完整性。

#### 4.2.1.3 决策支持阶段

数据仓库的建立就是要达到决策支持的目的。决策支持阶段包括信息查询和决策分析两个步骤。

数据仓库有两类用户，一类是信息查询者，他们是数据仓库的主要用户，他们用一种可预测的、重复性的方式使用数据仓库，达到他们的常规决策支持要求。另一类是知识探索者，他们是数据仓库的少量用户，他们用一种完全不可预测的非重复性的方式使用数据仓库，达到他们挖掘未知知识的要求，取得更大决策支持的效果。这两类不同的用户使数据仓库需要具有不同的性能或工具来满足他们的要求。

**1. 信息查询**

信息查询者使用数据仓库能发现目前存在的问题。例如，发现公司正在流失客户。

为适应信息查询者的要求，数据仓库一般采用如下方法提高信息查询效率：

(1) 创建数据陈列

对一些分散存放的不同物理位置的数据(如不同月份的数据)，创建一个数据陈列，将相关的数据(每月的数据)放在同一个物理位置。这样可以提高可预测的和有规律数据的查询效果。

(2) 预连接表格

对于两个或多个表格共享一个公用链或者共同使用的表格，可以将多个表格合并在一个物理表格中，提高数据的访问效率。

(3) 预聚集数据

利用“滚动概括”结构来组织数据。当数据输入到数据仓库时，以每天为基础存储数据。

在一周结束时，以每周为基础存储数据（即累加每天的数据）。月末时，则以每月为基础存储数据。通过这种方式来组织数据，可以极大地减少存储数据所需要的空间并潜在地提高性能。

（4）聚类数据

聚类将数据放置在同一地点，这样可以提高对聚类数据的查询。

**2. 知识探索**

知识探索者使用数据仓库能发现问题并找出原因。例如，找出流失客户的原因。

知识探索者通常用随意的、非重复的方式来查看大量的数据。为满足探索者对大量数据的需要，一般创建一个单独的探索仓库。这样，既不影响数据仓库的常规用户，又可以采用“标识技术”把数据压缩，放置在内存中，提高数据分析速度。

知识探索者一般使用一些模型来帮助决策分析，例如客户分段、欺诈监测、信用风险、客户生存期、渠道响应、推销响应等模型。通过模型的计算来得出一些有价值的商业知识。

知识探索者大量采用数据挖掘工具来获取商业知识。例如，通过数据挖掘得到如下一些知识：

- 哪些商品一起销售好？
- 哪些商业事务处理可能带有欺诈性？
- 高价值客户的共同点是什么？

知识探索者获取的知识为企业领导者提供决策支持，对保留客户、减少欺诈、提高公司利润具有重要作用。

#### 4.2.1.4 维护与评估阶段

该阶段包括数据仓库增长，数据仓库维护，数据仓库评估 3 个步骤。

**1. 数据仓库增长**

数据仓库建立以后，随着用户的不断增加以及时间的推移，用户查询需求更多，数据会迅速增长。造成这种增长的原因有：详细数据和汇总数据的增加，历史数据的增加；满足更多用户决策需求，数据的增加等。数据仓库在使用后不断增长已成为数据仓库的特点。

在数据仓库的开发过程中需要适应数据仓库不断增长的现实。

**2. 数据仓库维护**

数据仓库维护包括适应数据仓库增长的维护和正常系统维护两类。

适应数据仓库增长的维护包括数据增长的处理、存储空间的处理、ETL 处理、数据模型的修订、增强决策支持的处理等。其中，数据增长的处理工作包括去掉没有用的历史数据，以及根据用户使用的情况取消某些细节数据和无用的汇总数据，或增加实用的汇总数据。

存储空间的处理工作主要是对增长的存储设备要有计划。存储成本是软件成本的 4～5 倍。

正常的系统维护工作包括数据仓库的备份和恢复。由于数据仓库的数据是经过了复杂

的清洗和转换过程而得到的，因此它代表企业的丰富历史，它能适应用户信息查询和决策支持。备份数据内容是很有必要的。备份数据也为系统恢复提供基础，一旦系统出现灾难，利用备份数据可以迅速将数据仓库恢复到正常状态。

### 3. 数据仓库评估

数据仓库评估包括三个方面：系统性能评定；投资回报分析；数据质量评估。

（1）系统性能评定

它包括：

- 硬件平台是否能够支持大数据量的工作和多类用户、多种工具的大量需求？
- 软件平台是否是用一个高效的且优化的方式来组织和管理数据？
- 是否适应系统（数据和处理）的扩展？

（2）投资回报分析

投资回报分析包括定量分析和定性分析。

- 定量分析是计算投资回报率（ROI），即收益与成本的比率。IDC 公司提供的数据表明：欧美 62 家企业建立的数据仓库三年投资回报率平均值为 401%，收回投资的平均时间为 2.3 年。最终用户获得的效益大约占总效益的 50%，信息收集人员和维护人员获得的效益共占总效益的 50%。

IDC 的调查结果表明，对于环境比较复杂的企业，数据仓库是一种有价值的投资。

- 定性分析是分析如下几个方面的效果：企业与客户之间关系状态？给客户获得的好处？建立企业的合作关系如何？对转瞬即逝的机会快速反应能力如何？管理宏观和微观数据的能力如何？改善管理能力如何？

（3）数据质量评估

数据质量是数据仓库成功的关键，只有高质量的数据才能为决策支持提供准确的依据，保证决策的正确性。

数据质量的评估标准有：

① 数据是准确的。数据必须保证它的准确性，例如姓名、地址对营销部门必须正确。

② 数据符合它的类型要求和取值要求。定义了数据字段类型（如字符型、实数型等）后，对该字段的所有数据必须满足类型要求，其取值必须在指定的范围内，例如“性别”字段是“字符型”，其取值只有“男”或“女”。

③ 数据具有完整性和一致性。数据的完整性体现在对不同的需求。都应该获得所需要的数值，不应该有缺失值。数据的一致性体现在相同记录下同一字段的数据在多个不同的源系统中有相同的类型和取值，例如产品 ABC 的代码是 1234，在不同的源系统中都应该是一致的。

④ 数据是清晰的且符合商业规则。数据正确的命名可以帮助用户更好地理解数据元素，如果用户不了解它的含义就不可能很好地使用它。数据必须符合商业规则，例如销售价格不能低于底价，贷款余额不能是负值。

⑤ 数据保持时效性并不能出现异常。对不同时间要求的数据（如按照月）能按时提供，保持时效性。数据不能出现异常，例如客户的通讯地址不能是传真号码或者电话号码。

### 4.2.2 数据质量与数据清洗

数据质量是数据仓库的成功关键。完整而准确的数据能够大大提高客户服务的质量,例如,为产品增加交叉销售的机会(即购买一个产品时,可能购买其他产品)。高质量的数据能减少成本和降低风险,提高生产率,完成实时的信息分析,最本质的是保证战略决策的制定。

在数据仓库的开发中,数据的抽取和转换过程中会发现数据质量问题,要及时找出数据污染的原因,进行有效的数据清洗,确保数据的高质量。

#### 1. 数据质量问题

数据质量问题表现为:

(1) 字段中的虚假值。在输入数据时,有时会将字母 q、O 等,误改成数字“9”和“0”。

(2) 数据值缺失。这在客户数据中经常出现。

(3) 不一致的值。不同的源系统代码表示不一致。例如有的代码表示为 A(Auto)、H(Home)、F(Flood);有的表示为 1、2、3;有的表示为 AU、HO、FL 等。

(4) 违反常规的不正确值。例如一年工作的天数,加上假日、病假天数超过 365 天。

(5) 一个字段有多种用途。一个字段同一数据在不同部门可能有不同的含义。

(6) 标码不唯一。例如销售系统与库存系统的产品代码不一致。

#### 2. 数据污染产生的原因

出现数据被污染情况的原因有:

(1) 系统转换。由于系统升级而发生变化时,在文件转换过程中,会对数据产生污染。系统转换和迁移是数据污染的重要原因。查找数据污染需要了解每一次源系统所经过的转换过程。

(2) 数据老化。在源系统中有很多旧系统时,旧的值随着时间的变化会失去它的含义和意义,逐渐形成数据污染。

(3) 复杂的系统集成。数据不一致会产生数据污染。数据仓库的源系统种类越多,出现污染数据的可能性越大。

(4) 数据输入的不完整信息。在初始数据输入时,没有完全输入所有的字段,将导致数据值缺失;对必须输入的字段,随便输入一些通用数据,也将产生数据污染。

(5) 输入错误。错误的数据输入也是数据污染的一个主要来源。

(6) 欺诈。有些人为了欺诈,千方百计地往系统中输入错误的数据,特别是涉及金额或产品数量的字段。

(7) 缺乏相关政策。如果公司对数据质量没有明确的相关政策,它的数据质量就不可能得到保证。

#### 3. 数据清洗

清洗数据仓库中所有数据的成本是相当高的。在现实世界中,绝对的高质量数据是不存在的,不能期望 100%的数据质量。清洗数据采用“面向目标”的原则,先确定要使用哪些

数据,然后确定目标是什么。清洗数据要明确如下问题：

(1) 需要清洗哪些数据

清洗哪些数据是根据数据仓库要回答用户的问题类型,找出回答问题所需要的数据。权衡每部分数据的价值,并估计对数据清洗、对用户分析会造成什么影响。通常只清洗那些重要的数据,而忽略那些不重要的数据。

(2) 在什么地方清洗

数据的错误来自源系统,在数据进入数据仓库之后再进行清洗是不现实的,这样会破坏已转移和装载的其他数据。通常,数据在被存储进数据仓库之前就应该进行清洗。数据抽取过程中被抽取的数据一般进入缓存区域,数据装载过程从缓存区域进入数据仓库中。

在缓存区域中清洗数据相对容易。

(3) 怎么清洗

清洗源系统中的数据,必须找到适合源系统的字段和格式的清洗工具。现在已有很多完成各种数据清洗功能的工具软件可以采用。对于特殊的数据污染则要专门编制程序来完成数据清洗。

对于要净化的数据元素分为 3 个优先级类型：高优先级、中优先级和低优先级。对高优先级的数据要达到 100%的数据质量等级。中优先级的数据越准确越好,对这类数据,要在数据修正的成本和坏数据可能造成的影响之间进行平衡。低优先级的数据可以在有时间和有需要的时候进行清洗。

(4) 建立一个数据质量框架

数据质量框架包括：建立数据质量领导小组;建立数据质量政策和标准;定义质量指标参数和基准;识别受坏数据影响最大的商业功能;选择那些有较大影响力的数据元素,确定优先级;对有较大影响力的数据元素定制清洗计划,并执行数据清洗;再为较小影响的数据元素制定清洗计划,并执行数据清洗。这个框架是确保数据质量的基础。

### 4.2.3 数据粒度与维度建模

数据粒度是指数据仓库的数据中保存数据的细化程度或综合程度的级别。细化程度越高,粒度级别就越小;相反,细化程度越低,粒度级别就越高。

数据粒度深深影响存放在数据仓库中的数据量的大小,同时影响数据仓库所能回答的查询类型。

数据仓库的设计需要在数据量大小与查询的详细程度之间做出权衡。

例如,在数据仓库中存储一个顾客(张三)一个月里每个电话的细节,能够查询出“张三在某日是否给女友打过电话”。其存储量是每个月 200 个记录 40 000 个字节。若存储一个顾客一个月的电话综合,能够查询“张三这个月打了多少个长途电话”,其存储量是每个月一个记录 200 个字节。

#### 1. 大维度与雪花模型

大维度表现在两方面：①大维度表的记录数很大；②大维度表的属性很多。在数据仓库中,客户维度和产品维度是典型的大维度。一个全国连锁店的客户维度可能包括上亿条

记录之多。大型零售店的产品维度也相当巨大。

一般大型客户维度有2000万条记录,150个维度属性,可能有多种层次结构。

一般大型产品维度有100 000种产品,100多个属性,也有多种层次结构。

大维度数据仓库运行时会很慢,效率很低。

大维度表采用雪花模型的数据组织,是一种有效的方法。

对产品维度,产品分属于产品品牌,品牌又分属于产品分类;对客户维度,客户分属于地区,地区分属于国家。以上结构采用雪花模型的数据组织,将减少各维表的记录数,使查询过程中搜索记录数目减少。

对于销售的雪花模型如图4.11所示。

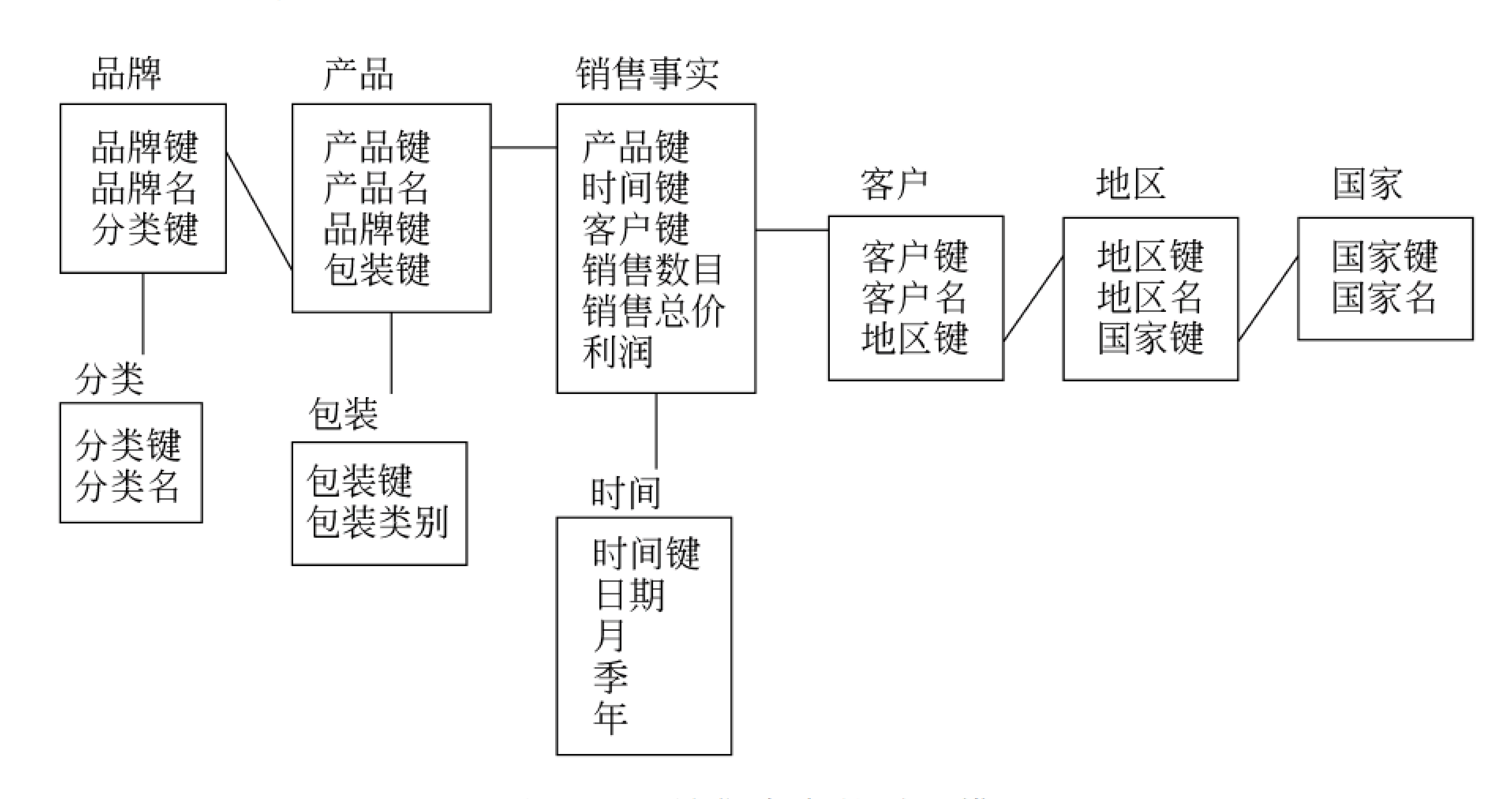



图4.11　销售事实的雪花模型

## 2. 综合事实表

在基础事实表中,各条记录反映维度多层结构中最低层次的数据。例如,销售事实是某日、某个商店和某个产品相关的销售数量和销售总价。

在现实中,大多数查询不是基于基础事实表来操作的,而是基于综合数据的查询。这样建立综合事实表是提高综合数据查询非常有效的方法,而且大大提高了数据仓库的性能。

在多维表中,很多维都具有层次结构,对不同维的层次的提升,将可建立多种综合事实表。综合事实表是由基础事实表衍生出来的。同时维度也将衍生出高层次的维表,它与综合事实表连接起来一起使用。

例如,对产品维从每一个具体的产品上升为分类产品,需要建立产品分类维表(衍生维表)。按照产品分类键来综合基础销售事实表的事实,形成综合销售事实表,如图4.12所示。

从图4.12中可见,对基础事实表查询利用产品维表,对综合事实表查询利用产品分类维表。

以上是对一个维度进行提升产生的综合事实表和衍生维表。若对两个或三个维度同时进行提升,所产生的综合事实表也需要衍生出相应二个或三个高层次的维表。综合事实表将大大提高综合数据的查询效果。

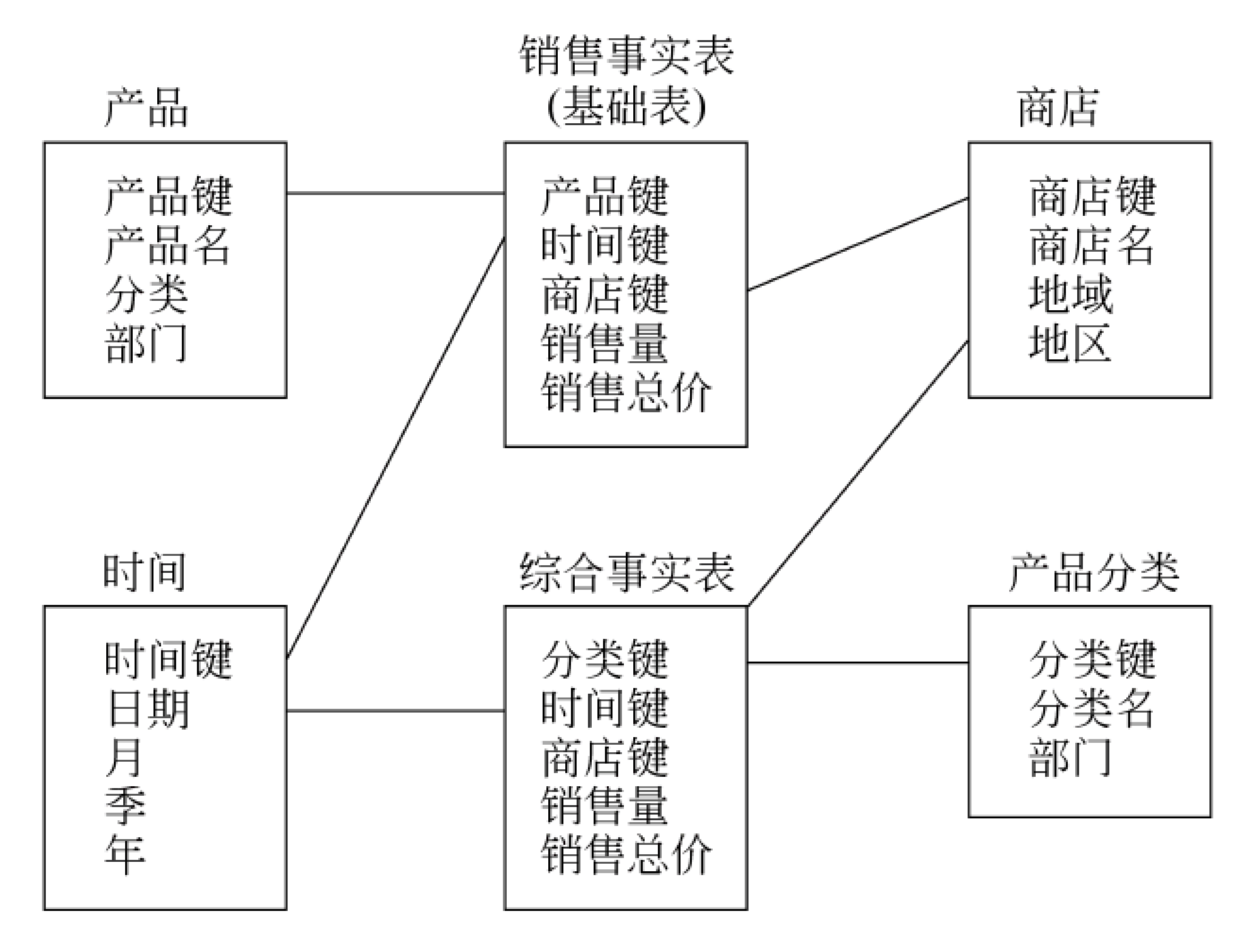



图 4.12　综合事实表和衍生维度(产品分类)表

# 4.3　数据仓库技术与开发的困难

## 4.3.1　数据仓库技术

数据仓库环境中的数据处理可以概括为装入与访问两个过程。数据从大量数据库中的集成、转换和装载到数据仓库中去。数据一旦被装入，通常是不更新的。数据到数据仓库后将被访问和分析。

### 1. 管理大量数据

对于数据仓库最重要的技术就是能够管理大量的数据。

传统数据库环境和数据仓库环境一个重要的区别在于，数据仓库中有更多的数据量，比一般的数据库环境中要多得多。数据仓库中的数据量是 10GB 或 100GB 级的，而一个通用的 DBMS 通常管理的数据是 MB 级的。数据仓库要管理大量的数据，是因为它们：

(1) 包括细节数据；

(2) 包括历史数据；

(3) 包括汇总数据；

(4) 包括元数据。

有很多种管理大量数据的方法——通过寻址，通过索引，通过数据的外延，通过有效的溢出管理等等。管理大量的数据有两方面：能够管理大量数据的能力和能够高效管理数据的能力。任何声称支持数据仓库的技术一定都要满足能力与效率的要求。数据仓库开发者建造数据仓库时，需要能够满足处理大量数据的需求。

### 2. 数据的高效装入和数据压缩

(1) 装入数据

数据仓库的一个重要技术就是能够高效地装入数据。

有很多种装入数据的方法：通过一个语言接口一次一条记录或者一起使用一个程序一次全都装入。另外，在装入数据的同时，索引也要高效地装入。在有些时候，为了平衡工作负载，数据索引的装入可以推迟。

如果数据仓库中数据的装入有不可克服的困难，那么这个数据仓库就没有用处了。

(2) 数据压缩

数据仓库的成功之处就在于能够管理大量的数据。达到这一目的的中心是数据的压缩。当数据能够被压缩时，它便能存储在很小的空间中。这与数据仓库的环境有关，因为数据在插入到数据仓库中后，是很少被更新的。数据仓库中数据的稳定性减少了空间管理问题，这些问题是在更新紧密压缩的数据时发生的。

压缩的另一个好处是程序员可以完全脱离给定的输入/输出操作。当然，对数据的访问就会有相应的解压缩的问题。虽然解压缩需要一定的开销，但这个开销不是 I/O 资源的开销，而是 CPU 的开销。通常，在数据仓库环境中 I/O 资源比 CPU 资源少得多，因此数据的解压缩并不是一个主要的问题。

### 3. 存储介质的管理

在处理大量数据时，为了满足高效率和合理的费用，应用在数据仓库中的基本技术应该能够解决多种存储介质的问题。仅仅在直接存取存储设备(如磁盘)上管理一个成熟的数据仓库是不够的。考虑到访问速度和存储费用，对数据的存储要分层次，层次的区分如下：

| 存 储 介 质 | 访 问 速 度 | 存 储 费 用 |
|---|---|---|
| 主存 | 非常快 | 非常贵 |
| 扩展内存 | 非常快 | 贵 |
| 高速缓存 | 非常快 | 贵 |
| 磁盘 | 快 | 适中 |
| 光盘 | 不慢 | 不贵 |
| 微缩胶片 | 慢 | 便宜 |

由于数据仓库中的大量数量和被访问到的可能性这两方面的因素存在，因此一个满载的数据仓库应该放在多种存储层次上。处理数据仓库技术应该能管理多种存储介质上的数据。

### 4. 元数据管理

数据仓库中的元数据比在传统的数据库中更重要。为了更加有效，数据仓库的用户应该能够对准确和实时的元数据进行访问。如果没有一个好的元数据来源进行运作的话，决策支持系统分析员的工作就非常困难。

### 5. 数据仓库语言

数据仓库需要有非常丰富的数据仓库语言。这种语言的作用是有效管理数据仓库中的

数据和快速、高效地访问数据仓库中的数据。3.4.5 节介绍的 MDX 语言就是一个有效的访问数据仓库语言。

**6. 高效索引**

数据仓库的灵魂就在于灵活性和对数据的不可预测的访问。这一点也就是要求能够对数据进行快速和方便的访问。数据仓库中的数据如果不能方便和有效地检索，那么建立数据仓库这项工作就不是成功的。当然，设计者可以利用许多方法来使数据尽可能的灵活，例如利用双重粒度级和数据分割。但这些技术一定要支持方便地索引，建立和应用索引的费用不能太高。

**7. 多维 DBMS 和数据仓库**

在数据仓库中经常讨论的技术是多维数据库管理系统(多维 DBMS)。多维数据库管理系统提供了一种信息系统结构，使得对数据的访问非常灵活，可以用多种方法对数据进行切片、分割，动态地考察汇总数据和细节数据的关系。多维 DBMS 不仅提供了灵活性，还可以对终端用户进行管理，这些非常适合决策支持系统(DSS)环境。为此，数据要定期从数据仓库中导入到多维 DBMS 中去。

数据仓库和多维 DBMS 的区别：

(1) 数据仓库有大量的数据；多维 DBMS 中的数据至少要少一个数量级。

(2) 数据仓库只适合于少量的灵活访问；而多维 DBMS 适合大量的非预知的数据的访问和分析。

(3) 数据仓库内存储了很长时间范围内的数据——从 5 年到 10 年；多维 DBMS 中存储着比较短时间范围内的数据。

(4) 数据仓库允许分析人员以受限的形式访问数据，而多维 DBMS 允许自由的访问。

多维 DBMS 和数据仓库有着互补的关系。数据仓库为非常细节的数据提供了基础，而这在多维 DBMS 中通常是不能看到的。数据仓库能容纳非常详细的数据，这些数据在导入到多维 DBMS 时被轻度综合了，导入多维 DBMS 后，数据还会被进一步地汇总。在这种模式下，多维 DBMS 可以包含除了非常细节以外的所有数据。使用多维 DBMS 的分析者可以用一种灵活和高效的方式来对多维 DBMS 中所有不同层次的数据进行钻取。如果需要的话，分析者还可以向下钻取到数据仓库。通过这种方式将数据仓库和多维 DBMS 结合。DSS 分析者可以得到这两者的好处。DSS 分析者大部分时间里可以在多维 DBMS 中享受其操作高效的优点，同时如果需要的话，还可以向下钻取最低层次的细节数据。

一些多维 DBMS 建立在关系模型上，而一些多维 DBMS 建立在能优化“切片和切块”数据的基础上，在这里数据可以被认为存储在多维立方体内，后者的技术基础为“数据立方体”。

两种技术基础都支持多维 DBMS 数据集市，但这两种技术基础之间存在着一些差异。

多维 DBMS(OLAP)是一种技术，而数据仓库是一种体系结构的基础。这两者之间存在着互补的和共生的关系。最一般的情况下，数据仓库作为多维 DBMS 的基础——从中选出细节数据的一个子集传到多维 DBMS 中，在那里，数据要么被汇总，要么被聚集。

### 4.3.2 数据仓库开发的困难

数据仓库由于数据量大(具有 GB 级到 TB 级的数据),数据包括近期、综合、历史等多个层次,还包括元数据,致使数据的存储和管理复杂。数据仓库的应用包括快速查询、多维分析及数据挖掘等多种类型。这样,数据仓库需要一个具有海量存储的硬件平台和一个能进行并行处理的大型数据库系统。大型数据库厂商 NCR 公司提供的数据仓库硬件平台是具有海量并行处理的 WordMark 系列服务器,数据仓库软件是 Teradata 数据库系统,能处理 GB 级到 TB 级的数据,具有很强的并行处理能力和扩展能力。ORACLE、IBM、SAS、Microsoft 等公司也都推出了各自的数据仓库商品,它们为开发数据仓库提供了强有力的工具。这些工具极大地推动了数据仓库的发展。但是,在国外仍存在开发数据仓库失败的案例。这些失败的案例主要反映在错误的认识观念上,它们构成了开发数据仓库的障碍。

国外总结开发数据仓库的典型错误归纳如下。

#### 1. 没有理解数据的价值

没有认识到数据的价值,就不会有效地访问数据和挖掘数据中的信息和知识。数据必须共享,才能充分发挥它的价值,那些垄断数据的做法只可能埋没数据的作用,直接影响数据仓库的开发。数据的一致性是数据共享的基础。数据对于不同的人,由于定义的不一致和时间的不一致,就会造成数据的不一致,这会造成对数据理解的不一致和报表的不一致,从而丧失人们对数据的信任,更谈不上辅助决策了。

#### 2. 未能理解数据仓库概念

不了解数据仓库的含义,即它所能解决的业务问题和它的用途,必然导致数据仓库开发的失败。数据仓库数据不是将大量现行系统中的数据堆积而成的。数据仓库是将现行管理系统中大量数据按决策主题重新组织,通过集成而形成的。数据仓库包含大量随时间变化的数据,而不进行实时更新。不像现行管理系统中数据进行实时更新,只保留当前准确的数据。在数据仓库中元数据很重要。元数据能够让用户了解数据仓库中有什么数据,它们是如何组织的,对这些数据如何使用。只有充分理解数据仓库的概念,才能充分发挥数据仓库作用。

#### 3. 尚未清楚了解用户将如何使用数据仓库之前,便贸然开发数据仓库

一个典型的错误观点是:“只要你建好(数据仓库)了,他们就会用”。这种盲目自信地建造数据仓库的做法,由于用户未参加界定对数据仓库的需求,必然导致数据仓库的失败。

数据仓库的建造必须要有用户代表参加。用户代表懂得数据仓库中需要有哪些数据,以及如何使用数据仓库来改善他们的决策过程。

#### 4. 对数据仓库规模的估计模糊

数据仓库规模包括数据量的多少、用户数量、常规查询所耗费的资源、并发查询数目、对

CPU 的要求等。

数据仓库中的数据量多少依赖于数据的主题(如顾客、产品、风险管理、收支等)的划分,以及用户人数。数据太多时,将会使数据存储和加载过程耗资巨大,还会造成数据得不到充分利用或根本无人使用它们的情况。

**5. 忽视了数据仓库体系结构和数据仓库开发方法**

数据仓库体系结构具有三个层次:数据获取、数据存储和分析工具。这个体系结构是建造数据仓库的图纸。

数据仓库的生命周期(DWLC)不同于系统生命周期(SDLC)。DWLC 包括系统分析与设计、数据获取、决策支持、维护与评估 4 个阶段 12 个步骤。

数据仓库的设计应该采用数据驱动方法,即以数据为基础(尽可能地利用已有的数据、代码等,而不是从无到有),进行从面向应用到面向分析需求的转变,按决策主题存取数据和分析数据,并逐步提高决策效果的方法。

数据仓库中数据必须保证它的质量,错误的数据会引起错误的决策。数据的粒度水平如何,即数据应该以细节形式存储,还是以概括形式存储,还是两种形式兼有,这应该根据用户需求来确定。

开发时只有克服了以上的错误观念,数据仓库才能真正发挥它的作用。

# 习　题　4

1. 数据仓库的需求分析的任务是什么?
2. 数据仓库系统需要确定的问题有哪些?
3. 实现决策支持所需要的数据包括哪些内容?
4. 什么是概念模型? 它的特点是什么?
5. E-R 图如何描述概念模型?
6. 比较数据库的概念模型设计与数据仓库的概念模型设计。
7. 解释图 4.1 所示的概念模型。
8. 什么是逻辑模型? 数据仓库的逻辑模型是什么?
9. 数据仓库的逻辑模型与数据库的逻辑模型有什么不同?
10. 举例说明从数据仓库的概念模型到逻辑模型的转换。
11. 在数据仓库中为什么要考虑数据的粒度层次划分?
12. 数据仓库的记录系统包含什么内容? 举例说明。
13. 什么是物理模型? 数据仓库的物理模型设计包括哪些工作?
14. 为什么数据仓库物理模型设计中要建立汇总计划和确定数据分区方案?
15. 说明图 4.8 中逻辑模型与物理模型的区别。
16. 概括说明“概念模型、逻辑模型、物理模型”分别是什么样的数据模型?
17. 数据仓库索引技术包括哪些内容?
18. 为什么 B-Tree 索引不适合数据仓库?

19. 数据仓库中采用标识技术有什么好处？

20. 数据仓库的广义索引是什么时候建立的(是在建立数据仓库之后，还是在建立数据仓库同时)？简单说明原因。

21. 说明数据仓库开发的 4 个阶段和 12 个步骤。

22. 简要说明数据仓库开发的分析与设计阶段的内容。

23. 简要说明数据仓库开发的数据获取阶段的内容。

24. 简要说明数据仓库开发的决策支持阶段的内容。

25. 简要说明数据仓库开发的维护与评估阶段的内容。

26. 数据质量问题表现在哪些方面？

27. 数据污染产生的原因有哪些？

28. 为什么大维度表采用雪花模型？

29. 数据仓库技术包括哪些内容？

30. 国外开发数据仓库的错误有哪些？

# 第5章　数据仓库的决策支持

## 5.1　数据仓库的用户

数据仓库的用户有两类：信息使用者和探索者。

信息使用者是使用数据仓库的大量用户。信息使用者以一种可预测的、重复性的方式使用数据仓库平台。他们通常查看概括数据或聚集数据，查看相同的商业维度（如产品、客户、时间）和指标（如收入和成本）随时间的发展趋势。他们天天重复同样的活动，很少使用元数据。他们的工作相对来说属于战术性的。

探索者完全不同于信息使用者，他们有一个完全不可预测的、非重复性的数据使用模式。探索者查看海量的详细数据，而概括数据则会妨碍探索者的数据分析。他们经常查看历史数据，而且查看历史数据的时间要比信息使用者长得多。探索者的任务是寻找公司数据内隐含的价值并且根据过去的事件努力预测未来决策的结果。探索者是典型的数据挖掘者。

### 5.1.1　数据仓库的信息使用者

信息使用者所提交的查询操作是均匀的且有相当少量的数据，他需要享有好的查询响应时间。数据仓库管理员采取如下方法来支持信息使用者的性能需求。

#### 1. 非规格化

数据建模和规范化的作用是产生一种完全没有数据冗余的设计方法。但是，有时在数据仓库设计中引入一些有限的数据冗余来提高数据访问效果。例如，在一些数据表中加入相同的量，这是用增加数据存储来换取数据访问的优化（减少查询时间）。

#### 2. 创建数据阵列

数据仓库管理员发现用户经常同时使用相关类型的数据时，应创建数据阵列，将这些数据单元存储在一起，提高访问效果。

例如，对于每年所有月份的数据，被分别放置在不同的物理位置上，而用户经常要同时查看1月、2月、3月等月份中的数据，这样会花费很多搜索时间到不同的物理位置去获取数据。一个好的方法是创建数据阵列，将相关联的数据放在同一物理位置，这样可以提高查询效率。

#### 3. 预连接表格

节省机器资源最有效的技巧之一，就是基于一个公用键和共同使用的数据将表格合并在一起。

例如，如果有两个或者更多的表格共享一个公用键，或者有相同的表格使用，那么可以将多个表格合并到一个物理表格中。这样做可以很大程度地提高数据访问效率。

#### 4. 预聚集数据

一种非常有用的方法是根据“滚动概括”结构来组织数据。

当数据被输入到数据仓库中时，以每小时为基础存储数据。在这一天结束时，以每天为基础存储累加每小时的数据。在一周结束时，以每周为基础存储累加每天的数据。月末时，则以每月为基础存储累加每周的数据。这样，在累加数据后，就删除被累加的细节数据，通过这种方式来组织数据，数据仓库管理者将极大地减少存储数据所需要的空间并潜在地提高性能。

当然，管理员也会丧失查看已过时的详细数据的能力，越早获取的数据，保留的详细数据越少。但是，许多种类型的数据可接收这种处理。例如，可以非常有效地积累销售、产品、市场数据。

#### 5. 聚类数据

在预测了用户使用需求以及使用规则后，将不同类型的数据并置在一起，即基于产生共同信息，将不同类型的数据记录放置在相同的物理位置。这使用户查看这些记录时可以在同一地点找到它们，提高了查询效率。

如果使用是不可预测和不规则的，那么数据聚类毫无意义。

#### 6. 压缩数据

压缩将节省资源，因为当系统访问一个物理数据块时，压缩将优化所检索的数据量。利用这种方式，压缩可以使可读取的数据量极大。但同时也需要用户有一定的经验。

要注意的是，在不需要数据更新时才可以使用数据压缩，即压缩需要不改变任何数据（即一旦写入，不允许重写或者更新）。

#### 7. 定期净化数据

数据仓库管理员通过定期删除数据仓库中不需要的数据，可以为每个用户提高性能。没有其他任何一种方法比删除不需要的数据对数据仓库更有利。

#### 8. 合并查询

如果查询定期发生，那么可以通过把这些查询合并到同一个表格中，来节省大量资源。查询合并的作用就是把扫描数据仓库表格的次数最小化。

合并查询功能的条件有：

- 当有多个查询询问相同的表格时；
- 所访问的表格是一个大表格；
- 用可预测的、有规律的方式来执行查询；
- 这些查询所执行的连接是一行接一行的方式；

- 这种查询对执行时间不太敏感。

如果查询不能够符合这些条件，合并查询功能就没有任何优势。

应该如何处理合并查询功能？数据仓库管理员收集所有查询需求并合并到一个大型池中。这些查询的焦点是某一个表格——主表格。只要与二级表格的连接经过某一行的连接点，这些查询就可以查看其他表格并作为查询处理的一部分。

从主表格开始，访问每一行。如果某一行符合任意一个查询的任何选择标准，则保留此行以被分析。否则继续执行下一行。一旦某行被证明令人感兴趣，则将所需要的数据写入到一个工作文件中。如果有多个查询都需要相同的数据（通过连接点），那么这个结果集将被多个查询所标记。

### 5.1.2 数据仓库的探索者

探索者是那些寻找不平常的且有用的商业运作模型的用户群。探索者的运作方式是反复无常的、不可预测的及随机的。大部分时间，探索者努力搜索但一无所获；偶尔探索者也会发现意外的、无价的信息“金块”。

探索者查看详细资料和历史记录。在多数情况下，探索者考虑数据的不同类型和数据具体值之间的关系。探索者要做的工作有概括分析、抽取、建模和分类。

#### 1. 概括分析

概括分析是探索者分析过程的第一步。探索者开始以分析数据仓库中数据的外部特征，即分析数据的完整性和准确性（数据质量）。在概括分析活动中，要询问的典型问题包括：

- 家庭收入如何分配？
- 有多少账户每月消费超过 200 元？有多少账户每月消费小于或等于 200 元？
- 有多少客户有两个以上的小孩并居住在市区？

#### 2. 抽取

通过概括分析，所选数据的轮廓已经基本显示出来了。数据抽取的任务就是从数据仓库中抽取指定的数据并组织起来，送入支持探索者分析的探索仓库中。这样，不会影响数据仓库的正常工作。

#### 3. 建模

探索者通过概括分析来理解数据，通过抽取来准备数据，通过建模来分析数据。

建模是开发一种用来描述实体（如客户、商品、渠道等）的关系模型的过程。探索者使用的模型有：

- 客户分段；
- 后续产品；
- 欺诈检测；
- 渠道响应（例如电话销售和直接邮寄）；

- 信用风险；
- 客户生存期价值；
- 推销响应。

例如，利用建模来确认有可能拖延支付电话账单的客户：首先，建立一个模型（利用统计学和行为科学）来确认经常拖延支付电话账单的客户特征。然后，根据客户与模型的密切程度，对所有的客户进行分类。这样，可以提供谁将不支付电话账单的某种可能性预测。最后，把那些与此模型有紧密关系的客户作为目标。

数据仓库管理员为保证探索者的有效工作，创建"探索仓库"很有必要。探索仓库是企业数据仓库的"转出"，用来支持某些特定的分析，也不妨碍企业数据仓库中其他常规用户的正常使用。

建立探索仓库所依赖的技术基础是基于"标识"的技术（参见 4.1.5 节中"2. 标识技术"），利用基于标识技术可使探索仓库非常经济。基于标识的技术允许把数据压缩到能将数据放置在内存中（全部或者大部分）的程度。一旦使用内存存储，分析和检索的速度将大大快于使用标准企业数据仓库时的速度。

探索仓库是临时性的、短期性的。探索仓库的特征是固定不变的构造和重建。一旦构造好某个探索仓库，则再也不需要构造具有同样形式或内容的探索仓库。探索仓库能够满足数据仓库环境中非结构化处理的需要。探索仓库适合于数据挖掘的探索者。

探索仓库一般使用规范化的数据结构，因为探索仓库适用于不知道自己需求的使用者。而星型模型的数据结构不适合探索仓库，因为星型模型需要在知道商业维度（如产品、客户、时间）和指标（如收入或者是成本）等的情况下使用数据。

元数据在探索仓库环境中也非常重要。因为探索者用多种方式查看探索仓库，所以元数据起到特别重要的作用。在探索仓库中，必须建立有效的元数据层。这个元数据层能够在每次重新构造探索仓库时被传输到探索仓库。

## 5.2 数据仓库的决策支持与决策支持系统

数据仓库是一种能够提供重要战略信息，并获得竞争优势的新技术，因此得到了迅速的发展。

经理们和管理者需要哪些战略信息来支持决策呢？例如，对自己公司的运营有全面深入的了解，了解关键因素和它们之间是如何相互作用的；监视这些因素是如何随时间变化的；将公司的运营状况和市场竞争及行业标准联系起来比较。经理们和管理者需要将注意力集中在客户的需求和喜好上，集中在新兴技术、销售、市场结果、产品和服务质量水平等事务上。制定和执行商业战略和目标时需要的信息类型应包含整个企业组织。

战略信息并不为企业日常运作所用，不是关于订货，发货，处理投诉或者从银行账户提款的信息。战略信息比这些信息重要得多，对于企业的生存和持续健康发展有非常重要的意义。企业决定性的商业决策有赖于正确的战略信息。

具体的战略信息有：

(1) 给出销售量最好的产品名单（排序）；

(2) 找出出现问题的地区(切片);

(3) 追踪查找出现问题原因(向下钻取);

(4) 对比其他的数据(横向钻取);

(5) 显示最大的利润;

(6) 当一个地区的销售低于目标值时,提出警告信息。

建立数据仓库的目的不仅是为了存储更多的数据,而且是要对这些数据进行处理并转换成商业信息和知识,利用这些信息和知识来支持企业进行正确的商业行动,并最终获得效益。

数据仓库的功能是在恰当的时间,把准确的信息传递给决策者,使他能做出正确的商业决策。

数据仓库的主要作用是帮助企业摆脱盲目性,提高决策的准确性和决策速度,也就是说,数据仓库的作用正是帮助企业把信息与知识转变为力量(实施正确的行动并获得效益)。

数据仓库的决策支持一般包括查询与报表、多维分析与原因分析、预测未来。NCR 数据仓库公司提出了动态数据仓库及相应的决策支持:实时决策和自动决策。

针对实际问题,利用决策支持能力,通过人机交互,达到辅助决策的系统称为决策支持系统。

## 5.2.1 查询与报表

查询与报表是数据仓库的最基本、使用得最多的决策支持方式。查询与报表可以使决策者了解“目前发生了什么”。

### 1. 查询

数据仓库提供的查询环境的特点是:

(1) 能向用户提供查询的初始化、公式表示和结果显示等功能。

(2) 由元数据来引导查询过程。

(3) 用户能够轻松地浏览数据结构。

(4) 信息是用户自己主动索取的,而不是数据仓库强加给他们的。

(5) 查询环境必须要灵活地适应不同类型的用户。

查询服务具体体现为:

(1) 查询定义。确保数据仓库用户能够容易地将商业需求转换成适当的查询语句。

(2) 查询简化。让数据和查询公式的复杂性对用户透明。让用户能够简单地查看数据的结构和属性。使组合表格和结构简单易用。

(3) 查询重建。有些简单的查询也能导致高强度的数据检索和操作,因此要使用户输入的查询进行分解并重新塑造,使其能更高效地工作。

(4) 导航的简单性。用户能够使用元数据在数据仓库中浏览数据,并能容易地用商业术语而不是技术术语来导航。

(5) 查询执行。使用户能够在没有任何 IT 人员的帮助下执行查询。

(6) 结果显示。能够以各种方法显示查询结果。

(7) 对聚合的了解。查询过程机制必须知道聚合的事实表,并且在必要的时候能够将查询重新定义到聚合表格上,以加快检索速度。

### 2. 报表

大部分查询均要以报表形式输出。数据仓库构建的报表环境有:

(1) 预格式化报表。提供这些报表清晰的描述说明。使用户能够容易地浏览格式化报表库中的报表并选择他们需要的报表。

(2) 参数驱动的预定义报表。与预格式化的报表相比,参数驱动的预定义报表给了用户更大的灵活性。用户必须有能力来设置它们自己的参数,用预定义格式创建报表。

(3) 简单的报表开发。当用户除了与格式化报表或预定义报表外还需要新的报表时,他们必须能够轻松地利用报表语言撰写工具来开发他们自己的报表。

(4) 公布和订阅。数据仓库设置选项让用户公布他们自己创建的报表,并允许其他用户订阅或者接收这些报表的拷贝。

(5) 传递选项。提供各种选项,诸如群发、电子邮件、网页和自动传真等让用户传递报表,允许用户选择他们自己的方式来接收报表。

(6) 多数据操作选项。用户可以请求获得计算出来的指标,通过交换行和列变量来实现结果的旋转,在结果中增加小计和最后的总计,以及改变结果的排列顺序等操作。

(7) 多种展现方式选项。提供多种类型的选项,包括图表、表格、柱形格式、字体、风格、大小和地图等。

## 5.2.2 多维分析与原因分析

多维分析与原因分析能让决策者了解"为什么会发生"。

### 1. 多维分析

多维分析是数据仓库的重要的决策支持手段。数据仓库中心数据是以多维数据存储的。通过多维分析将获得在各种不同维度下的实际商业活动值(如销售量等),特别是他们的变化值和差值,以达到辅助决策效果。例如通过多维分析得到如下信息:

- 今年以来,公司的哪些产品量是最有利润的?
- 最有利润的产品是不是和去年一样?
- 公司今年这个季度的运营和去年相比情况如何?
- 哪些类别的客户是最忠诚的?

这些问题的答案是典型的基于分析的面向决策的信息。决策分析往往是事先不可知的。例如,一个经理可能会以查询品牌利润按地区的分布情况来开始他的分析活动。每一个利润的数值指的是,在指定时间内,某个品牌所有产品在该地区的所有地方销售利润的平均值。每一个利润数值都可能是由成千上万的原始数据汇聚而成的。

这些分析都是在多维数据分析的基础之上进行的。

**2. 原因分析**

查找问题出现的原因是一项很重要的决策支持任务，一般通过多维数据分析的钻取操作来完成。

例如，某公司从分析报表中得知最近几个月来整个企业的利润在急速下滑，为此系统分析员利用数据仓库的原因分析的决策支持手段，通过人机交互找出该企业利润下滑的原因。具体步骤如下：

(1) 查询整个公司最近3个月来各个月份的销售额和利润，通过检索数据仓库中的数据显示销售额正常，但利润下降。

(2) 查询全世界各个区域每个月的销售额和利润，通过检索多维数据和切块，显示欧洲地区销售额下降，利润急剧下降，其他地区正常。

(3) 查询欧洲各国销售额和利润。通过对多维数据的钻取，显示一些国家利润率上升，一些国家持平，欧盟国家利润率急剧下降。

(4) 查询欧盟国家中的直接和间接成本。通过对多维数据的钻取，得出欧盟国家的直接成本没有问题，但间接成本提高了。

(5) 查询间接成本的详细情况。通过钻取查看详细数据，得出企业征收了额外附加税，使利润下降。

通过原因分析，得出企业利润下滑的真正原因是欧盟国家征收了额外附加税。

在数据仓库中，在宏观数据的切片中发现的问题，通过向下钻取操作，查看下层大量详细的多维数据，才能发现问题出现的原因。针对具体问题，通过数据仓库的原因分析，找出问题发生的原因的过程，这是一个典型的数据仓库决策支持系统简例。

### 5.2.3 预测未来

预测未来使决策者了解“将要发生什么”。

数据仓库中存放了大量的历史数据，从历史数据中找出变化规律，将可以用来预测未来。在进行预测的时候需要用到一些预测模型。最常用的预测方法是采用回归模型，包括线性回归或非线性回归。利用历史数据建立回归方程，该方程代表了沿时间变化的发展规律。预测时，将预测的时间代入到回归方程中去就能得到预测值。一般的预测模型有多元回归模型、三次平滑预测模型、生长曲线预测模型等。

除用预测模型外，采用聚类模型或分类模型也能达到一定的预测效果。

聚类模型是对没有类的大量实例，利用距离的远近(如欧式距离和海明距离等)，把大量的实例聚成不同的类，如K-means聚类算法和神经网络的Kohonen算法等。把实例聚完类后，对新的例子，仍用距离大小来判别它属于哪个类。

对于分类模型，它是对已经有了类别后，分别对各个不同类进行类特征的描述，如决策树方法、神经网络的BP模型等。分类模型是通过对各类实例的学习后，得到各类的判别知识(即决策树、神经网络的网络权数值等)，利用这些知识可以对新例判别它属于哪个类别。

### 5.2.4 实时决策

数据仓库的第 4 种决策支持是企业需要准确了解“正在发生什么”，从而需要建立动态数据仓库(实时数据库)，用于支持战术型决策，即实时决策，有效地解决当前的实际问题。

完成第 1 到第 3 种决策支持的数据仓库都以支持企业内部战略性决策为重点，帮助企业制定发展战略。数据仓库对战略性的决策支持是为企业长期决策提供必须的信息，包括市场划分、产品(类别)管理战略、获利性分析、预测和其他信息。

战术性决策支持的重点则在企业外部，支持的是执行公司战略的员工。第 4 种侧重于战术性决策支持。

数据仓库的“实时决策”是指为现场提供信息实时支持决策，如能及时补给的库存管理和包裹发运的日程安排及路径选择等。许多零售商都倾向于由卖主管理库存，自己则拥有一条零售链和众多作为伙伴的供货厂商，其目的是通过更有效的供货链管理来降低库存成本。为了使这种合作获得成功，就必须向供货商详细地提供有关销售、促销推广、库内存货等信息的知情权。之后便可以根据每个商店和每个商品对库存的要求，建立并实施有效的生产和交货计划。为了保证信息确实有价值，必须随时刷新信息，还要非常迅速地对查询做出响应。

动态数据仓库能够逐项产品、逐个店铺、逐秒地做出最佳决策支持。

以货运为例，统筹安排货运车辆和运输路线，需要进行非常复杂的决策。卡车上的货物常常需要打开，把某些货物从一辆车转移到另一辆车上，以便最终送抵各自的目的地。这有些像旅客在枢纽机场转机。当某些卡车晚点时，就要做出艰难的决定：是让后继的运输车等待迟到的货物，还是让其按时出发。如果后继车辆按时出发而未等待迟到的包裹，那么迟到包裹的服务等级就会大打折扣。反过来说，等待迟到的包裹则将损害后继的运输车上的其他待运包裹的服务等级。

运输车究竟等待多长时间，取决于需卸装到该车辆的所有延迟货物的服务等级和已经装载到该车辆的货物的服务等级。很显然，第二天就应该抵达目的地的货物和数天后才需达目的地的货物，二者的服务等级及其实现难度是大不相同的。此外，发货方和收货方也是决策的重要考虑因素。对企业盈利十分重要的客户，其货物的服务等级应该相应地提高，以免因货物迟到破坏双方的关系。延误货物的运输路线、天气条件和许多其他的因素也应予以考虑。能够在这种情况下做出明智的决策，相当于解决了一个非常复杂的优化问题。

显而易见，零担散货部经理应在先进决策支持功能的帮助之下，极大地提高其计划和路径选择的决策质量。更重要的是，若要实现数据仓库的决策支持能力，作为决策基础的信息就必须保持随时更新。这就是说，为了使数据仓库的决策功能真正服务于日常业务，就必须连续不断地获取数据并将其填充到数据仓库中。战略决策可使用按月或周更新的数据，而以这种频率更新的数据是无法支持战术决策的。此外，查询响应时间必须以秒为单位来衡量，才能满足作业现场的决策需要。

与传统的数据仓库一样，最佳的动态数据仓库是跨越企业职能和部门界限的。它既可为战术决策也可为战略决策提供资源支持。动态数据仓库是为支持企业级业务目标而设计的。与传统的数据仓库相比，它更加深入到企业内部，能将企业的多种渠道，包括网络、呼叫

中心和其他客户联络点联为一体。它还意味着通过网络,在企业各个角落配置决策人员。

动态数据仓库的主要功能是缩短重要业务决策及其实施之间的时间。重要的是将动态数据仓库所做的数据分析转换成可操作的决策,这样才能将数据仓库的价值最大化。动态数据仓库的主导思想是提高业务决策的速度和准确性,其目标是达到近乎实时决策,生成最大价值。

### 5.2.5 自动决策

数据仓库的第5种决策支持是由事件触发,利用动态数据仓库自动决策,达到"希望发生什么"。

动态数据仓库在决策支持领域中的角色越重要,企业实现决策自动化的积极性就越高。在人工操作效果不明显时,为了寻求决策的有效性和连续性,企业就会趋向于采取自动决策。在电子商务模式中,面对客户与网站的互动,企业只能选择自动决策。网站中或ATM系统所采用的交互式客户关系管理(CRM)是一个个性化产品供应、定价和内容发送的优化客户关系的决策过程。这一复杂的过程在无人介入的情况下自动发生,响应时间以秒或毫秒计。

随着技术的进步,越来越多的决策由事件触发,自动发生。例如,零售业正面临电子货架标签的技术突破。该技术的出现废除了原先沿用已久的手工更换的老式聚酯薄膜标签。电子标签可以通过计算机远程控制来改变标价,无需任何手工操作。电子货架标签技术结合动态数据仓库,可以帮助企业按照自己的意愿,实现复杂的价格管理自动化;对于库存过大的季节性货物,这两项技术会自动实施复杂的降价策略,以便以最低的损耗售出最多的存货。降价决策在手工定价时代是一种非常复杂的操作,往往代价高昂,超过了企业的承受能力。带有促销信息和动态定价功能的电子货架标签,为价格管理带来了一个全新的世界。而且,动态数据仓库还允许用户采用事件触发和复杂决策支持功能,以最佳方案,逐件货品、逐家店铺、随时做出决策。在CRM环境中,利用动态数据仓库,根据每一位客户的情况做出决策都是可能的。

激烈的竞争形势和日新月异的技术革新推动了决策技术的进步。动态数据仓库可以为整个企业提供信息和决策支持,而不只限于战略决策过程。然而,战术决策支持并不能代替战略决策支持。确切地说,动态数据仓库同时支持这两种方式。动态数据仓库的主要工作量仍然是战略性的。

### 5.2.6 决策支持系统

数据仓库整合了企业的各种信息来源,能确保一致与正确详细的数据。它是一个庞大的数据资源。要将数据转换成商业智能,就需要利用数据仓库来建立决策支持系统。

基于数据仓库的决策支持系统是针对实际问题,利用分析工具或者编制程序,采用一种或多种组合的决策支持能力,例如随机查询,灵活的报表,预测模型等,对数据仓库中的数据进行多维分析,从而掌握企业的经营现状,找出现状的原因,并预测未来发展趋势,弥补经验和直觉的不足,协助企业制定决策增强竞争优势。

根据NCR公司在企业政策制定调查中,发现企业的决策危机日益严重。虽然有更多

的数据,但是也有更多的决策,同时决策也更加复杂化。

调查中有98%的管理者说数据一直在增加,随着数据每年两倍或三倍地增长,他们会被数据“淹没”。有75%的管理者表示他们每天所做的决策比以往多。有52%的决策更为复杂,这其中有83%的人说他们必须针对每一决策去参考三个或更多的信息来源。

只有建立基于数据仓库的决策支持系统,才能适应这种发展趋势,才能在适当的时间获得正确的信息,快速地将这些信息转换成正确的决策。

NCR公司总裁M. Hard列举了三个不同性质公司失败的案例是不明智决策的结果。

(1) 霸菱银行,英国最老的银行之一(成立于1762年),在1995年因为在新加坡分公司一位员工有2.9万美元的错误,在伦敦的管理层,并不清楚在新加坡所发生的状况,由于在决策上历经一连串错误的决策,不出三年,银行垮了。分析原因,霸菱银行缺乏企业单一整合的观点,缺乏可用详细的数据,显然在每日,每周甚至于每年的基准上,缺乏适当的检查点或事业监督。

(2) F. W. Woolworth于1879年在美洲开了第一家店,118年来它提供了优惠价格的产品,培养了广大的客户忠诚度。它一直是人们采购商品的地方,可以买到任何东西。但是,他忽略了人口统计的改变与人们搬住郊区的趋势,未实时随市场的改变而调整,最终被崭新的零售业,如Wal-Mart与Target等公司击败。

(3) 美国环球航空TWA,1920年开始航空邮递时代,1930年,它在现代技术进展上领先,曾提供横贯大陆与横贯大西洋的飞行。但是,后来它缺乏信息科技的基础建设来应付新的竞争环境,在多处还停留在30年前技术的基础建设上。在倒闭前一年,终于了解必须结合来自多个系统的财务、市场与销售数据,以因应市场改变快速且做出精确的反应,但一切都为时以晚。

对以上的三个公司的分析得出,建立基于数据仓库的决策支持系统就可以避免失败的命运。

## 5.3 数据仓库应用实例

### 5.3.1 航空公司数据仓库决策支持系统简例

#### 1. 航空公司数据仓库系统的功能

航空公司数据仓库功能模块有:

市场分析:分析国内、国际、地区航线上的各项生产指标;

航班分析:分析某个特定市场上所有航班的生产情况;

班期分析:分析某个特定市场上各班期的旅客、货运分布情况;

时段分析:分析一段时间范围内每天不同时段的流量分布;

效益分析:分析航线、航班的效益;

机型分析:分析不同种机型对客座率等关键指标的影响;

因素分析:分析某个关键指标发生变化后对其他指标的影响程度。

### 2. 数据仓库系统的决策支持

利用数据仓库系统提供的决策支持有：

- 一段时间内某特定市场占有率、同期比较、增长趋势；
- 各条航线的收益分析；
- 计划完成情况；
- 流量、流向分析；
- 航线上各项生产指标变化趋势的分析；
- 航线上按班期分析、汇总各项趋势；
- 航线上按航班时刻分析各项指标；
- 航线上不同航班性质比较；
- 航线上运力投入结构比较；
- 分机型的航线运输统计；
- 飞机利用率统计；
- 城市对流量、流向对比；
- 航向分机型收益比较；
- 航班计划评估；
- 航线上不同机型的舱位利用情况。

### 3. 决策支持系统简例

通过查询"北京到各地区的航空市场情况"，发现西南地区总周转量出现了最大负增长量。该决策支持系统简例就是完成对此问题进行多维分析和原因分析，找出原因。

具体步骤如下：

(1) 查询：全国各地区的航空总周转量并比较去年同期状况

从数据仓库的综合数据中，查出北京到国内各地区航空周转量并与去年同期比较增长量，制成直方图进行显示，如图 5.1 所示。

从图 5.1 中看到从北京到国内各地区的总周转量以及与去年同期的比较情况，发现"北京—西南地区"出现的负增长最大。

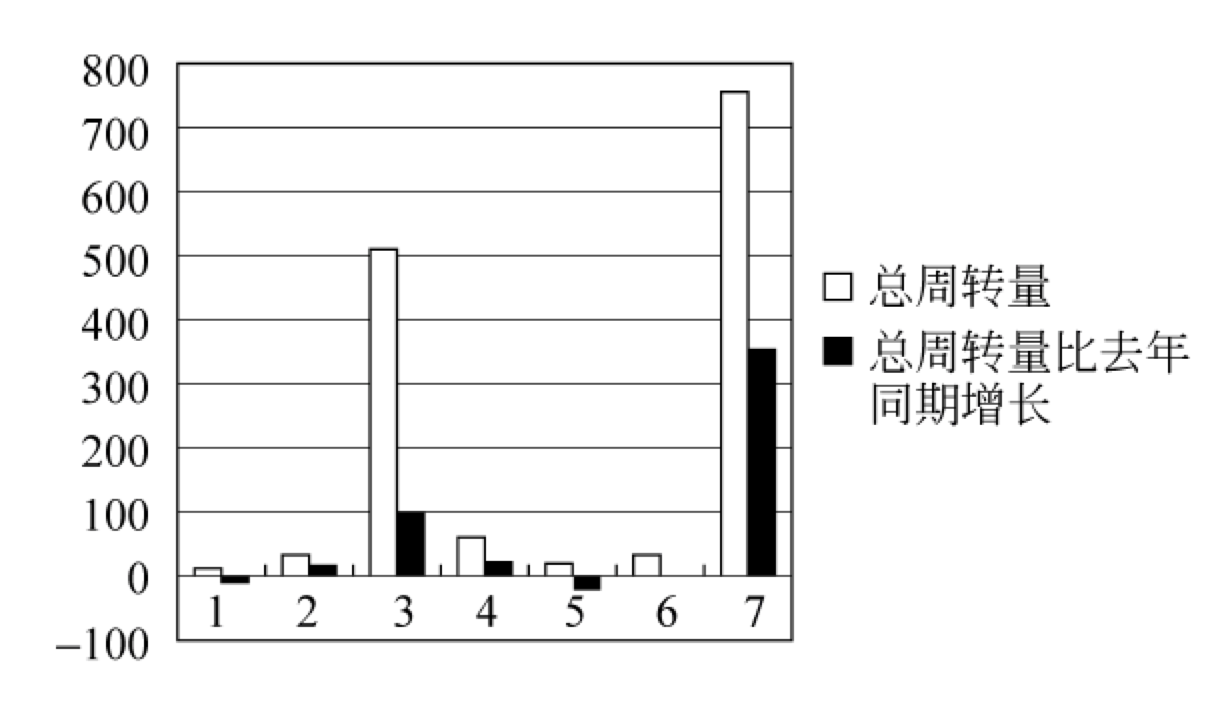
(2) 查询：全国各地区客运周转量以及和去年同期相比较

从数据仓库的总周转量数据中下钻到客运周转量并与去年同期比较增长量，制成直方图显示，如图 5.2 所示。

从图 5.2 中看到客运周转量及与去年同期比较，西南地区负增长在全国是最大的，其次是东北地区。

(3) 查询：全国各地区航空货运周转量及与去年同期比较

从数据仓库的总周转量数据中下钻到货运周转量并与去年同期比较增长量，制成直方图显示，如图 5.3 所示。




(说明：1：东北地区；2：华北地区；3：华东地区；4：西北地区；5：西南地区；6：新疆地区；7：中南地区)

图 5.1　全国各地区航空总周转量与去年对比状况

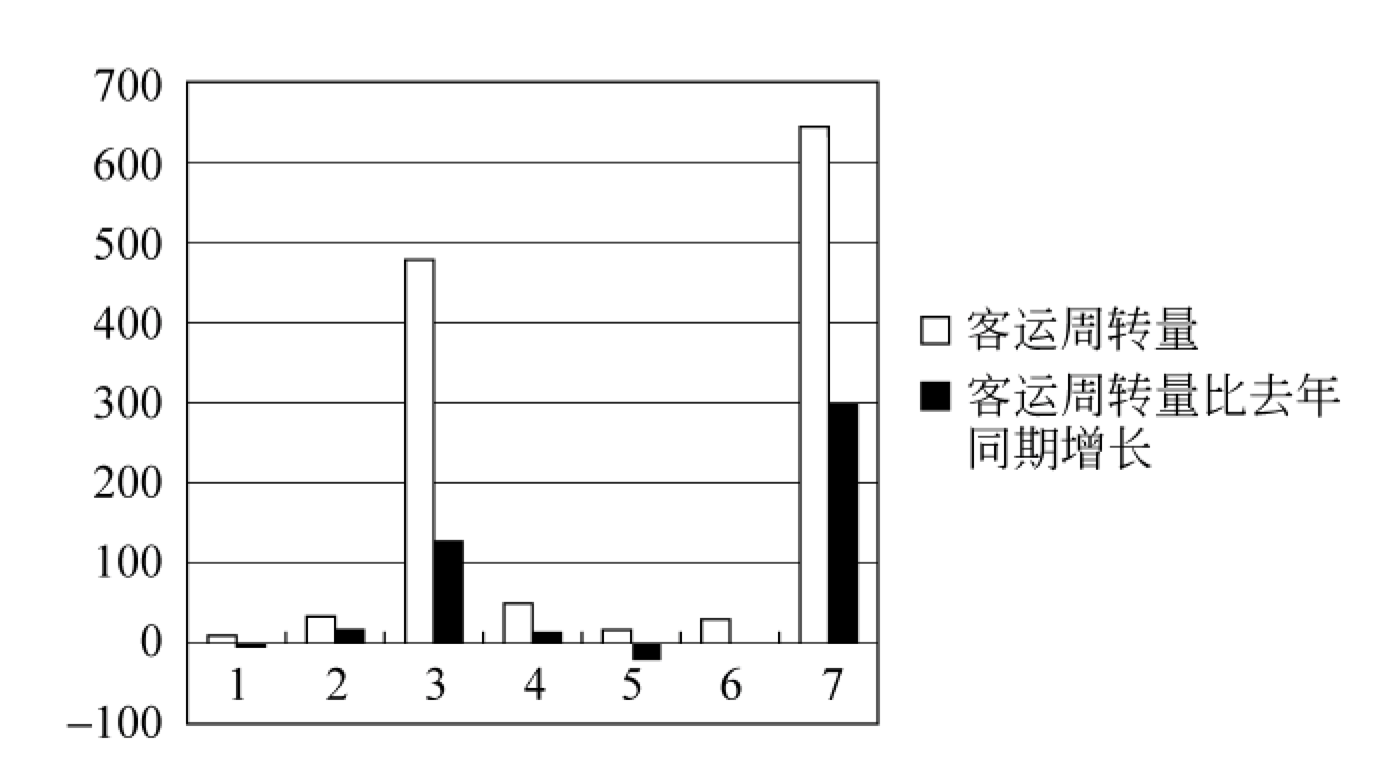



图 5.2　全国各地区航空客运周转量及与去年同期比较

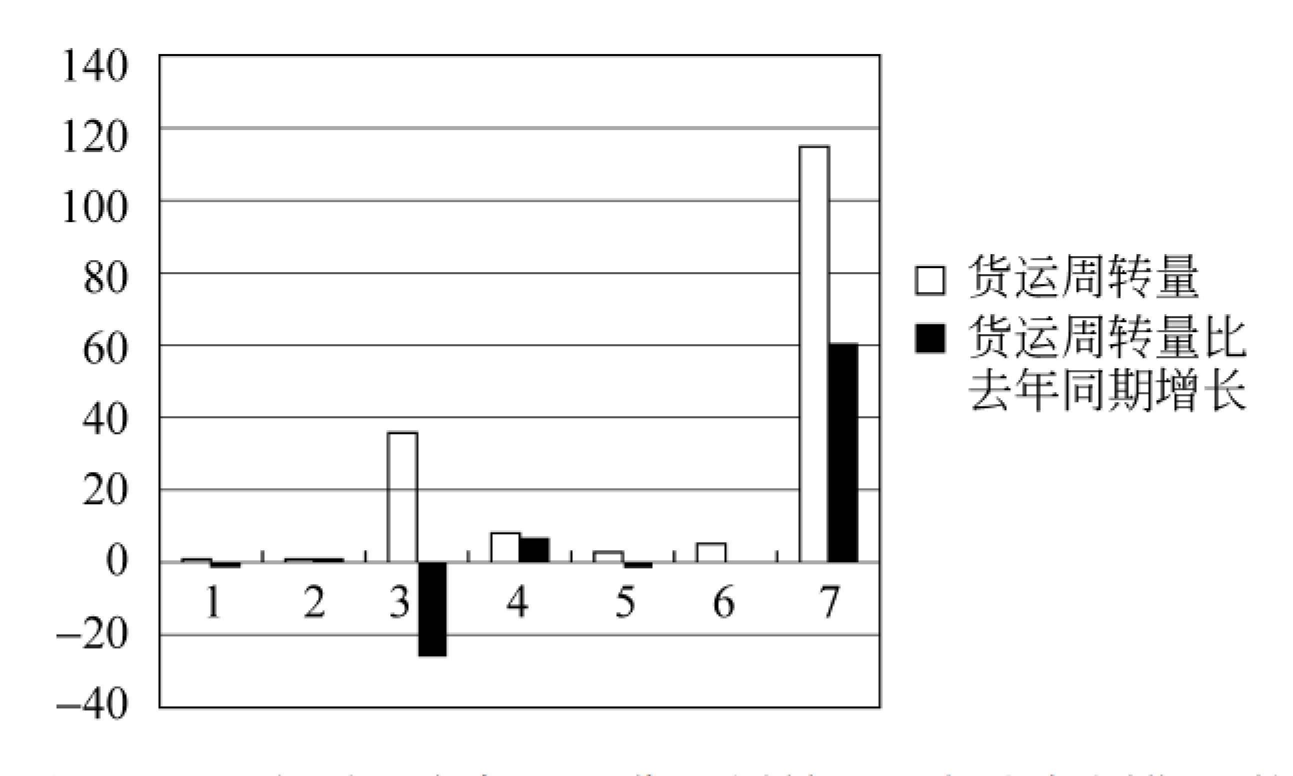



图 5.3　北京到国内各地区货运周转量及与去年同期比较

从图 5.3 中看到货运周转量及与去年同期比较，华东地区负增长在全国是最大的，西南地区也有负增长。

(4) 查询：全国各地区客运、货运、总周转量及其去年同期比较的具体数据

从数据仓库综合数据中直接取数据，制成表格显示，如表 5.1 所示。

从表 5.1 中可以看出航空客运、货运、总周转量以及与去年同期比较的具体数据。西南地区总周转量的负增长主要是客运负增长为主体。

表 5.1　客运、货运、总周转量及其去年同期比较

| | 客运周转量 | 对比去年增长量 | 货运周转量 | 对比去年增长量 | 总周转量 | 对比去年增长量 |
|---|---|---|---|---|---|---|
| 东北地区 | 11.86 | －5.1 | 1.29 | －1.5 | 13.15 | －6.6 |
| 华北地区 | 34.88 | 15.03 | 1.11 | 0.75 | 36 | 15.78 |
| 华东地区 | 479.30 | 126.52 | 36.16 | －25.59 | 515.46 | 100.93 |
| 西北地区 | 51.60 | 18.05 | 9.0 | 7.2 | 60.6 | 25.25 |
| 西南地区 | 15.43 | －19.35 | 3.29 | －0.56 | 18.72 | －19.91 |
| 新疆地区 | 29.02 | 0 | 5.85 | 0 | 34.87 | 0 |
| 中南地区 | 643.43 | 295.86 | 116.85 | 60.70 | 760.28 | 356.56 |

（5）查询：西南地区昆明、重庆两地航空总周转量以及与去年同期比较

从数据仓库总周转量下钻到西南地区昆明、重庆两地的总周转量以及与去年同期的比较，制成直方图显示，如图 5.4 所示。

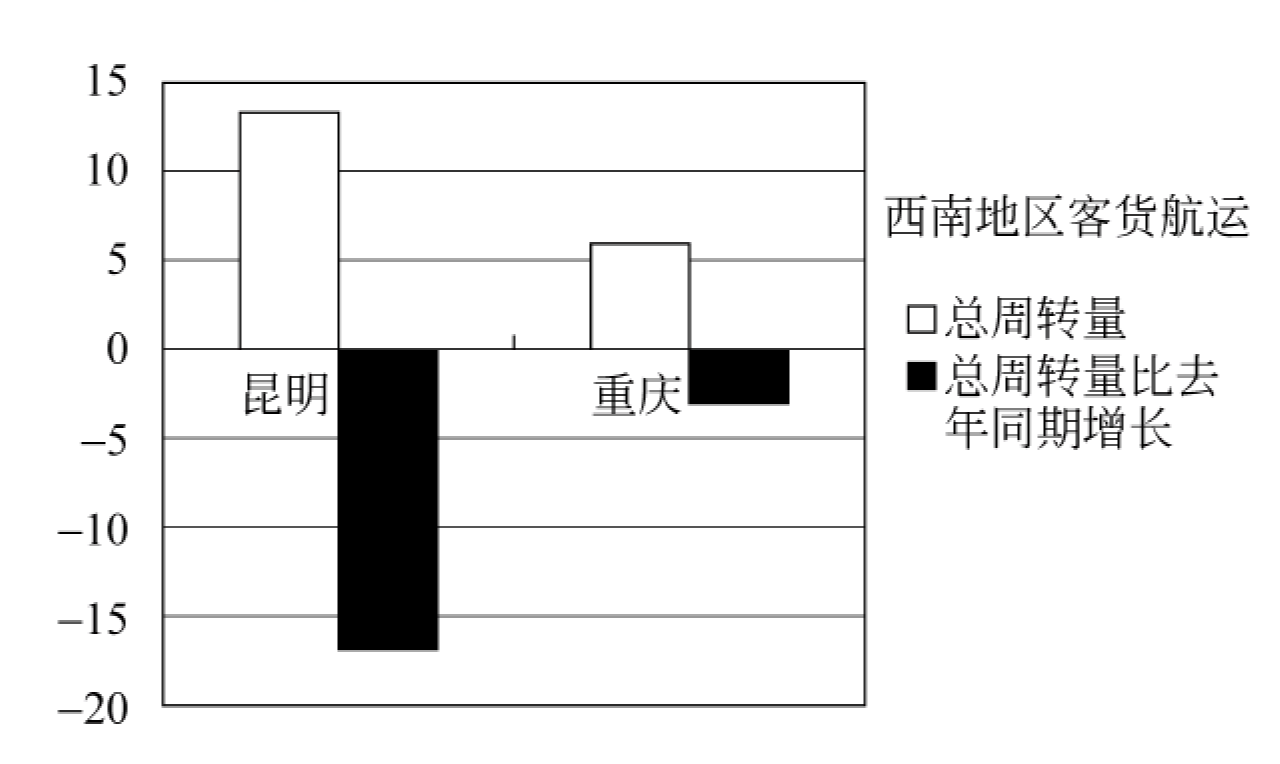
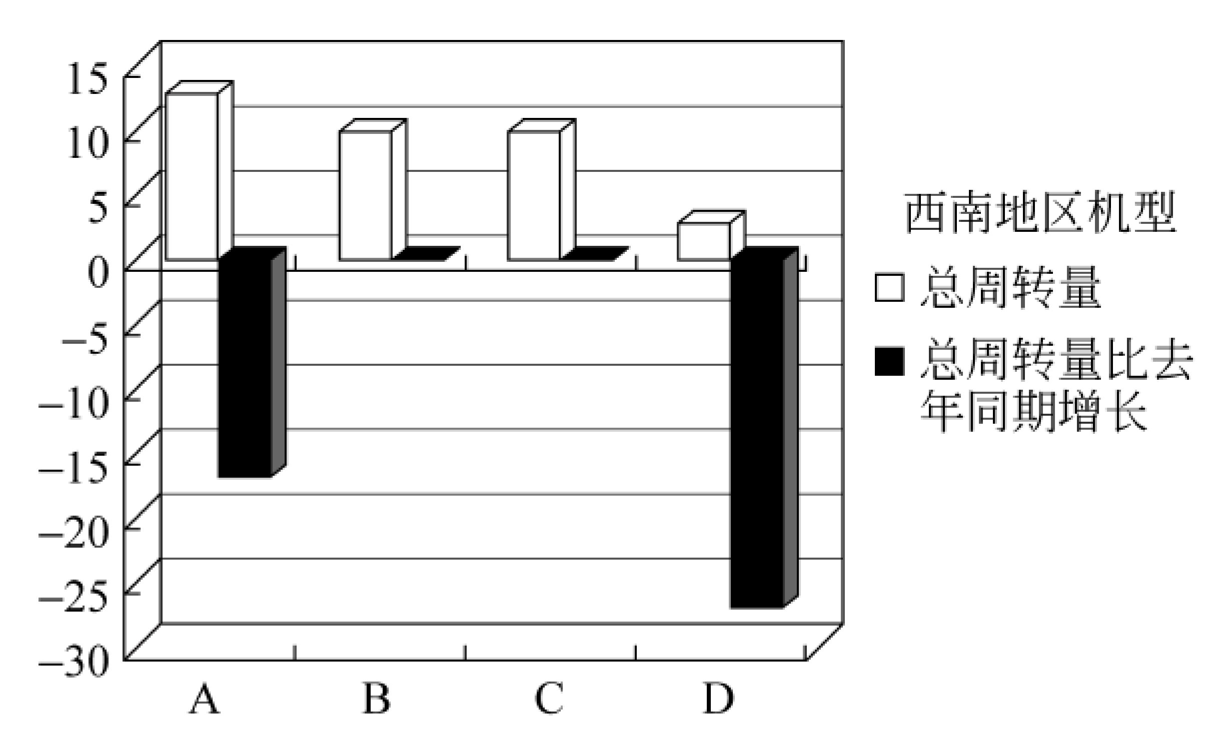



图 5.4　西南地区昆明、重庆两地航空总周转量及与去年同期比较

从图 5.4 中看出，西南地区航空总周转量下降最多的是昆明航线。

（6）查询：昆明航线按不同机型显示各自的总周转量并比较去年同期情况

从数据仓库中西南地区取出按机型维的各自机型的总周转量以及比较去年同期增长量，用柱形图显示，如图 5.5 所示。




(说明：A：150座级；B：200座级；C：300座级以上；D：200~300座级)

图 5.5　昆明航线各机型总周转量以及与去年同期比较的柱形图

从图 5.5 中可以看出昆明航线中 200～300 座级机型负增长最大，其次是 150 座级机型也有较大的负增长，而 200 座级以及 300 座级以上机型保持同去年相同的航运水平。

（7）查询：昆明航线按不同机型的周转量并比较去年同期的具体数据

从数据仓库中直接取数据，制成表格显示，如表 5.2 所示。

**表 5.2　昆明航线各机型总周转量以及与去年同期比较的数据**

| | 总周转量 | 对比去年增长量 |
|---|---|---|
| 150 座级 | 12.99 | －16.83 |
| 200 座级 | 10.07 | 0 |
| 300 座级以上 | 10.07 | 0 |
| 200～300 座级 | 2.91 | －26.9 |

从表 5.2 中可以看出不同机型的总周转量以及对比去年同期增长的具体数据。

以上决策支持系统过程完成了对航空公司全国各地区总周转量对比去年同期出现负增长量最大的西南地区，经过多维分析和原因分析，找出其原因发生在昆明航线上，主要是 200～300 座级机型的总周转量负增长以及 150 座级机型负增长量造成的。其中，200～300 座级负增长最严重。这为决策者提供了解决西南地区负增长问题辅助决策的信息。

#### 4. 决策支持系统结构图

将以上决策支持系统过程用决策支持系统结构图画出，如图 5.6 所示。

#### 5. 决策支持系统应用

以上决策支持系统只是找出西南地区航运负增长问题是由在昆明航线上 200～300 座级以及 150 座级机型的负增长所直接造成的。还可以通过昆明航线上航班时间以及其他方面进行原因分析，找出其他原因，为决策者提供更多的辅助决策信息。

同样，可以根据国内各地区航空市场状况对比去年同期增长显著的中南地区，找出总周转量大幅提高的原因。

从正反两方面来进行多维分析和原因分析，将可以得到更多的辅助决策信息，减少负增长，增大正增长，提高更大利润。

进行多方面分析的大型决策支持系统，将可以发挥更大的辅助决策效果。

### 5.3.2　统计业数据仓库系统

#### 1. 统计业数据仓库解决方案

统计信息是科学决策和宏观管理的重要基础，是国民经济核算的中心，是了解国情国力、指导国民经济和社会发展的信息主体。统计部门作为国家法定的专职信息职能部门，担负着对国民经济和社会发展情况进行统计调查、统计分析、提供统计资料和统计咨询意见、实行统计监督的神圣职责。

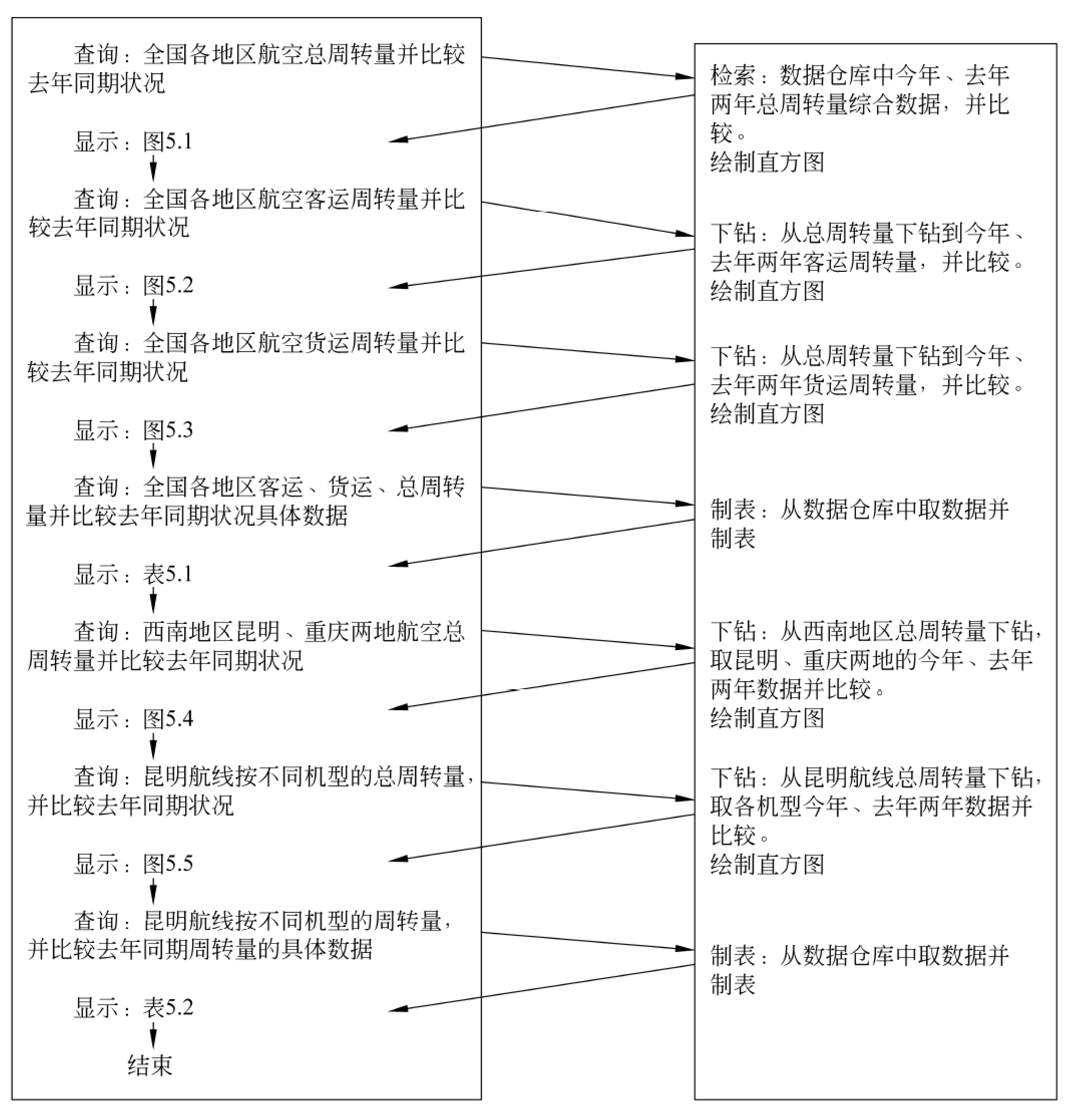



图 5.6　决策支持系统结构图

目前，国外统计行业成功的做法之一是采用先进的、成熟的数据仓库技术。数据仓库是信息技术领域的新概念，是近年来迅速发展起来的一种信息存储及管理技术。它存储大量的、决策分析所必需的、历史的、分散的各种数据，经过处理将这些资料和数据转换成集中统一、随时可用的信息。它能方便地提供统计业务人员和各级领导进行随机查询和任意的分析处理；它具有在任何时间、任何业务、回答任何问题的能力；利用数据仓库前端的数据挖掘工具和人工智能技术，统计业务人员还可以建立各种统计调查、统计分析和统计预测模型，以分析国民经济、工农业产值、人口等领域的现状及发展变化趋势和方向。

利用数据仓库技术，既能够快速实现传统的统计报表、统计图形功能，又能够利用数据仓库的数据挖掘技术在统计预测和决策支持管理中发挥重要作用。

面对日新月异的信息技术，统计业面临以下三方面的需求：

(1) 数据的集中存储与管理

统计行业掌握着大量的、各历史年度的原始调查资料，受历史和技术(数据库存储处理能力的限制)等因素的制约，这些资料大都还保留在纸介质、脱机的磁带和软盘上。由于缺乏大型数据库的集中存储和统一管理，随着年代的增加，这些资料的保存和安全受到严峻的考验；同时；这些宝贵的原始资料不能为统计业务人员随机查询和充分共享，不能进行有效的统计分析、预测评估和使用；难以快速地为管理决策提供科学依据。

(2) 查询方式和分析手段的更新

随着统计数据处理方式由逐级汇总到计算机超级汇总的转变，统计报表和统计分析需要从大量各种各样的原始材料中汇总整理各种不同需求，反映不同侧面的综合分析数据，传统的处理手段主要通过编写程序来实现；这样做的模式是固定的，且维护工作量大，开发周期长。为改变这种现状就需要一种技术或一种前端查询分析工具，统计业务人员可以根据任意条件、任意模式进行任意组合生成查询结果，同时利用该工具能进行分析处理，能够方便地组成各种多维报表和统计图形，如条形图、饼图、曲线图、多维立方图等。另外，针对一些深层次的研究需要，还应提供一些统计分析智能软件和智能算法以预测未来经济发展模式和走势。

(3) 与 Web 技术的有机结合

数据仓库技术与 Web 技术结合起来是采用目前流行的三层应用体系结构对系统进行应用开发。所谓三层结构，是指后台是数据仓库，前台是 Web 服务器，分布在各地的客户端采用浏览器的应用模式。利用这种技术，可以实现网上动态信息发布、网上随机查询和网上联机分析处理等功能，最终的目标是使统计业务人员的日常工作完全在 Web 上实现。

针对以上需求，信息领域新技术的应用特别是数据仓库技术的应用，是必然趋势。

**2. 某市统计局企业微观数据仓库系统**

实现某市统计局企业微观数据仓库是把掌握的不同专业、不同时期、分散的企业微观数据信息，按照多个主题集中存储和管理在数据仓库中，灵活地、非常方便地实现固定的和随机动态的数据查询处理、综合分析和统计报表。根据统计信息自动化总体规划要求，这些查询、分析和报表功能以及今后统计人员的日常业务处理工作都需在 Web 上进行。

在实现数据仓库之前，某市统计局已开发有企业微观数据库系统，受当时技术条件的限制，该系统的设计思路是按工业、建筑业、运输邮电业和批发零售贸易、餐饮业等不同专业分别建模，每个专业都对应的一套数据存储表和管理字典，共性数据依照专业被进行分割、分别进行存储，这样做虽然数据管理条理清楚，安全性能好，查询方式易于接受，但存在的问题是查询方式不够灵活，不同专业的指标横向比较困难，难以实现产、供、销等企业生产各个阶段数据的一条龙分析研究，同时受软件条件限制，无法实现 Web 方式查询且速度较慢。数据仓库是面向主题建模，在进行设计的时候，将企业微观数据仓库设计成以下主题：

(1) 企业基本情况：各年度、各专业统计调查单位基本情况名录的主要内容及全部标识性内容。

(2) 企业财务状况：各年度、各专业企业的资产、经营投入、产出效益等财务经营状况。

(3) 企业劳动状况：各年度、各专业企业的就业人数及工资收入情况。

(4) 企业消耗状况：各年度、各专业企业生产所需的原材料及能源消耗情况，包括价值量和实物量消耗情况。

(5) 企业生产状况：各年度、各专业企业的主营生产情况。由于不同专业的生产方式不同，又下设了若干子方面及工业产品产销存情况，建筑业生产完成情况，公路、水运、港口企业生产完成情况，商业、餐饮业销售经营情况。

这样建模以后，不同年度、不同专业的同类数据被集中进行存储，如此一来，指标无论是横向比较还是纵向比较都非常容易，并且整个系统只需要维护一套数据字典(元数据)。

数据建模是数据仓库设计中非常重要的一个环节，它包括逻辑建模和物理建模。在企业微观数据仓库中是利用 ERWIN 专业工具来建立模型，并形成相应的数据库结构。企业微观数据仓库的源数据是历年存储到微机上的数据，数据的格式、存储方式不尽相同，在加载到数据仓库之前，这些数据必须经过净化筛选、加工整理以及数据集成。利用 NCR 提供的 FastLoad 和其他工具，能方便地将经过处理的数据加载到 NCR 数据仓库里。目前企业微观数据仓库已存储 2 年各 4 个专业的历史数据，其他年度的数据正在整理当中。

应用开发的模式是基于目前流行的三层结构，即后台是数据仓库，前台是 Web 服务器，客户端是浏览器。Brio Enterprise 商业智能工具提供了很好的基于 Web 浏览器的查询、联机分析及报表功能，并且具有极高的安全性和严格的权限访问等级。企业微观数据仓库系统的前端应用都是基于 Web 方式开发，它具有网上随机查询、网上多维分析、网上数据钻取、网上图形分析、网上表格旋转透视、网上多维报表等功能，并且操作方式都是拖拉方式，今后统计业务人员的月报、年报等数据处理都可以在网上进行。数据仓库的好处、效益和威力被发挥得淋漓尽致。

### 5.3.3 沃尔玛数据仓库系统

美国的沃尔玛(Wal-Mart)是世界最大的零售商。2002 年 4 月，该公司跃居《财富》500 强企业排行第一。在全球拥有 4000 多家分店和连锁店。Wal-Mart 建立了基于 NCR Teradata 数据仓库的决策支持系统，它是世界上第二大的数据仓库系统，总容量达到 170TB 以上。

沃尔玛成功的重要因素是与其充分利用了信息技术分不开的，也可以说是对信息技术的成功运用造就了沃尔玛。强大的数据仓库系统将世界 4000 多家分店的每一笔业务数据汇总到一起，让决策者能够在很短的时间里获得准确和及时的信息，并做出正确和有效的经营决策。而沃尔玛的员工也可以随时访问数据仓库，以获得所需的信息，而这并不会影响数据仓库的正常运转。关于这一点，沃尔玛的创始人山姆・沃尔顿在他的自传《Made in America: My Story》一书是这样描述的："你知道，我总是喜欢尽快得到那些数据、我们越快得到那些信息、我们就能越快据此采取行动，这个系统已经成为我们的一个重要工具"。沃尔玛的数据仓库始建于 20 世纪 80 年代。自 1980 年以来，NCR 一直在帮助沃尔玛经营世界上最大的数据仓库系统。1988 年沃尔玛数据仓库容量为 12GB，1989 年升级为 24GB，以后逐年增长，1996 年其数据量达 7.5TB，1997 年为了圣诞节的市场预测和分析，沃尔玛将数据仓库容量扩展到 24TB。而到了信息技术飞速发展的今天，沃尔玛的数据仓库已经惊人地达到了超过 170TB。利用数据仓库，沃尔玛对商品进行市场类组分析(Marketing

Basket Analysis)，即分析哪些商品顾客最有希望一起购买。沃尔玛数据仓库里集中了各个商店一年多详细的原始交易数据。在这些原始交易数据的基础上，沃尔玛利用自动数据挖掘工具(模式识别软件)对这些数据进行分析和挖掘。一个意外的发现就是：跟尿布一起购买最多的商品竟是啤酒！按常规思维，尿布与啤酒风马牛不相及，若不是借助于数据仓库系统，商家绝不可能发现隐藏在背后的事实：原来美国的太太们常叮嘱她们的丈夫下班后为小孩买尿布，而丈夫们在买尿布后又随手带回了两瓶啤酒。既然尿布与啤酒一起购买的机会最多，沃尔玛就在它的一个个商店里将它们并排摆放在一起，结果是尿布与啤酒的销量双双增长。由于这个故事的传奇和出人意料，所以一直被业界和商界所传诵。

这个故事仅仅是沃尔玛借助数据仓库受益的一连串成功故事的一个花絮而已。如今，沃尔玛利用 NCR 的 Teradata 对超过 7.5TB 的数据进行存储，这些数据主要包括各个商店前端设备(POS，扫描仪)采集来的原始销售数据和各个商店的库存数据。Teradata 数据库里存有 196 亿条记录，每天要处理并更新 2 亿条记录，要对来自 6000 多个用户的 48 000 条查询语句进行处理。销售数据、库存数据每天夜间从 4000 多个商店自动采集过来，并通过卫星线路传到总部的数据仓库里。沃尔玛数据仓库里最大的一张表格(Table)容量已超过 300GB，存有 50 亿条记录，可容纳 65 个星期 4000 多个商店的销售数据，而每个商店有 5 万～8 万个商品品种。利用数据仓库，沃尔玛在商品分组布局、降低库存成本、了解销售全局、进行市场分析和趋势分析等方面进行决策支持分析，具体表现如下。

### 1. 商品分组布局

作为微观销售的一种策略，合理的商品布局能节省顾客的购买时间，能刺激顾客的购买欲望。沃尔玛利用前面提到的市场类组分析(MBA)，分析顾客的购买习惯，掌握不同商品一起购买的概率，甚至考虑购买者在商店里所穿行的路线、购买时间和地点，从而确定商品的最佳布局。

### 2. 降低库存成本

加快资金周转，降低库存成本是所有零售商面临的一个重要问题。沃尔玛通过数据仓库系统，将成千上万种商品的销售数据和库存数据集中起来，通过数据分析，以决定对各个商店各色货物进行增减，确保正确的库存。数十年来，沃尔玛的经营哲学是“代销”供应商的商品，也就是说，在顾客付款之前，供应商是不会拿到货款的。NCR 的 Teradata 数据仓库使他们的工作更具成效。数据仓库强大的决策支持系统每周要处理 25000 个复杂查询，其中很大一部分来自供应商，库存信息和商品销售预测信息通过电子数据交换(EDI)直接送到供应商那里。数据仓库系统不仅使沃尔玛省去了商业中介，还把定期补充库存的担子转嫁到供应商身上。1996 年，沃尔玛开始通过 Web 站点销售商品，商品都是从供应商处直接订货。Web 站点销售相当成功，在其投入运营的第一个周末就卖出了 100 多万件商品。

### 3. 了解销售全局

各个商店在传送数据之前，先对数据进行如下分组：商品种类、销售数量、商店地点、价格和日期等。通过这些分类信息，沃尔玛能对每个商店的情况有个细致的了解。在最后一

家商店关门后一个半小时，沃尔玛已确切知道当天的运营和财政情况。凭借对瞬间信息的随时捕捉，沃尔玛对销售的每一点增长，库存货物百分比的每点上升和通过削价而提高的每一份销售额都了如指掌。

**4. 市场分析**

沃尔玛利用数据挖掘工具和统计模型对数据仓库的数据仔细研究，以分析顾客的购买习惯、广告成功率和其他战略性的信息。沃尔玛在每周六的高级会议上要对世界范围内销售量最大的 15 种商品进行分析，然后确保在准确的时间、合适的地点有所需要的库存。

**5. 趋势分析**

沃尔玛利用数据仓库对商品品种和库存的趋势进行分析，以选定需要补充的商品，研究顾客购买趋势，分析季节性购买模式，确定降价商品，并对其数量和运作做出反应。为了能够预测出季节性销售量，它要检索数据仓库拥有 100 000 种商品一年多来的销售数据，并在此基础上进行分析和知识挖掘。

山姆·沃尔顿在他的自传中写道："我能顷刻之间把信息提取出来，而且是所有的数据。我能拿出我想要的任何东西，并确切地讲出我们卖了多少。"这感觉就像在信息的海洋里，"轻舟已过万重山"。他还写道："我想我们总是知道那些信息赋予你一定的力量，而我们能在计算机内取出这些数据的程度会使我们具有强大的竞争优势。"

沃尔玛神奇的增长在很大部分也可以归功于成功地建立了基于 NCR Teradata 的数据仓库系统。数据仓库改变了沃尔玛，而沃尔玛改变了零售业。在它的影响下，世界顶尖零售企业 Sears、Kmart、JCPenney、No. 1GermanRetailer、日本西武、日本三越等先后建立了数据仓库系统。沃尔玛的成功给人以启示：唯有站在信息巨人的肩头，才能掌握无限，创造辉煌。

# 习　题　5

1. 数据仓库两类用户有什么本质的不同？
2. 数据仓库的信息使用者与数据库的信息使用者有什么不同？
3. 信息使用者的主要性能需求是什么？
4. 为什么要增加数据冗余能提高查询速度？
5. 聚集数据与聚类数据有什么不同？
6. 什么是合并查询？
7. 探索者所做的工作有哪些？
8. 数据仓库的探索者的工作与数据库的数据挖掘者的工作有什么不同？
9. 说明企业需要哪些战略信息与实现方法。
10. 简述数据仓库查询服务内容。
11. 说明数据仓库的查询内容与数据库的查询内容有什么不同。
12. 说明如何利用数据仓库发现问题并找出产生问题的原因。

13. 说明如何利用数据仓库来进行预测。

14. 数据仓库如何实现实时决策？

15. 数据仓库如何实现自动决策？

16. 对 5.2.2 节中原因分析的实例，设计并画出决策支持系统结构图。

17. 在国内某市统计局数据仓库中选出两个主题画出星型模型图。

18. 利用沃尔玛数据仓库系统说明数据仓库的价值。

19. 利用数据仓库的数据资源建立的决策支持系统与传统的利用模型资源和数据库的数据资源建立的决策支持系统有什么区别？如何合并起来建立具有更强能力的决策支持系统？

# 第6章　数据挖掘原理

## 6.1　数据挖掘综述

### 6.1.1　数据挖掘与知识发现

知识发现(Knowledge Discovery in Database，KDD)被认为是从数据中发现有用知识的整个过程。数据挖掘被认为是KDD过程中的一个特定步骤，它用专门算法从数据中抽取模式(pattern)。

KDD过程定义为(Fayyad，Piatetsky-Shapiror 和 Smyth，1996)：KDD是从数据集中识别出有效的、新颖的、潜在有用的，以及最终可理解的模式的高级处理过程。

其中，数据集：事实 $F$(数据库元组)的集合；模式：用语言L表示的表达式E，它所描述的数据是集合 $F$ 的一个子集 $F_E$，它是 $F_E$ 的精练表达，我们称E为模式；有效、新颖、潜在有用、可被人理解：表示发现的模式有一定的可信度，应该是新的，将来有实用价值，能被用户所理解。

KDD过程图如图6.1所示。

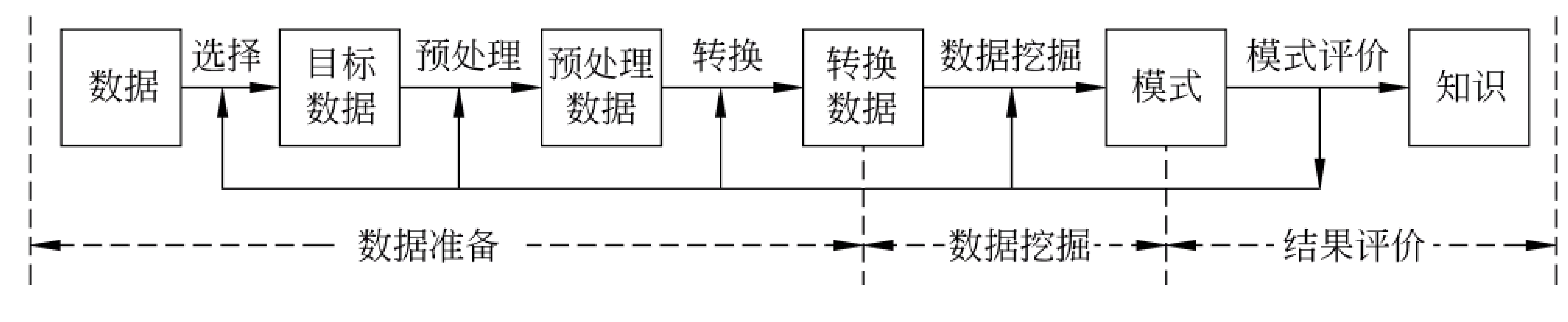



图6.1　KDD过程图

KDD过程可以概括为三部分：数据准备(Data Preparation)、数据挖掘(Data Mining)及结果的解释和评价(Interpretation & Evaluation)。

**1. 数据准备**

数据准备又可分为三个子步骤：数据选择(Data Selection)、数据预处理(Data Preprocessing)和数据转换(Data Transformation)。

数据选择的目的是确定发现任务的操作对象，即目标数据(Target Data)，是根据用户的需要从原始数据库中选取的一组数据。数据预处理一般包括消除噪声、推导或计算缺值数据、消除重复记录等。数据转换的主要目的是完成数据类型转换(如把连续值数据转换为离散型数据，以便于符号归纳，或是把离散型数据转换为连续值型数据，以便于神经网络计算)，尽量消减数据维数或降维(Dimension Reduction)，即从初始属性中找出真正有用的属性以减少数据挖掘时要考虑的属性的个数。

### 2. 数据挖掘

数据挖掘是利用一系列方法或算法从数据中获取知识。按照数据挖掘任务的不同,数据挖掘方法分为聚类、分类、关联规则发现等。聚类方法是在没有类别的数据中,按"距离"的远近聚集成若干类别,典型的方法有 K-means 聚类方法。分类方法是对有类别的数据,找出各类别的描述知识,典型的方法有 ID3、C4.5、IBLE 等分类方法。关联规则发现是对多个数据项重复出现的概率,超过指定的阈值时,建立这些数据项之间的关联规则,典型的方法有 Agrawal 提出的关联规则挖掘方法等。

利用数据挖掘方法获得的知识,是对这些数据的高度浓缩。

### 3. 结果的解释和评价

数据挖掘阶段获取的模式,经过评价,可能存在冗余或无关的模式,这时需要将其剔除;也有可能模式不满足用户要求,这时则需要回退到发现过程的前面阶段,如重新选取数据、采用新的数据变换方法、设定新的参数值,甚至换一种挖掘算法等。另外,KDD 由于最终是面向人类用户的,因此可能要对发现的模式进行可视化,或者把结果转换为用户易懂的另一种表示,如把分类决策树转换为 if…then…规则。

数据挖掘仅仅是整个过程中的一个步骤。数据挖掘质量的好坏有两个影响要素:一是所采用的数据挖掘技术的有效性,二是用于挖掘的数据的质量和数量(数据量的大小)。如果选择了错误的数据或不适当的属性,或对数据进行了不适当的转换,则挖掘的结果是不会好的。

整个挖掘过程是一个不断反馈的过程。比如,用户在挖掘途中发现选择的数据不太好,或使用的挖掘技术产生不了期望的结果。这时,用户需要重复先前的过程,甚至从头重新开始。

可视化技术在数据挖掘的各个阶段都扮演着重要的角色。特别是在数据准备阶段,用户可能要使用散点图、直方图等统计可视化技术来显示有关数据,以期对数据有一个初步的了解,从而为更好地选取数据打下基础。在数据挖掘阶段,用户则要使用与领域问题有关的可视化工具。在表示结果阶段,则可能要用到可视化技术,以使得发现的知识更易于理解。

## 6.1.2 数据挖掘对象

数据挖掘的对象主要是关系数据库和数据仓库,这是典型的结构化数据。随着技术的发展,数据挖掘对象逐步扩大到半结构化或非结构化数据,这主要是文本数据、图像和视频数据以及 Web 数据等。

### 1. 关系数据库

目前,建立的数据库都是关系数据库,数据仓库的数据存储仍然是关系数据库。数据挖掘方法也主要是研究数据库中属性之间的关系,挖掘出多个属性取值之间的规则。由于关系数据库的特点,促使了数据挖掘方法的改善。数据库的特点如下。

(1) 数据动态性

数据的动态变化是数据库的一个主要特点。由于数据的存取和修改,使数据的内容经

常发生变化，这就要求数据挖掘方法能适应这种变化。渐增式数据挖掘方法就是针对数据变化后，挖掘的规则知识能满足变化后的数据库内容。

(2) 数据不完全性

这主要反映在数据库中记录的域值丢失或不存在(空值)。这种不完全数据给数据挖掘带来了困难。为此，必须对数据进行预处理，填补该数据域的可能值。

(3) 数据噪声

由于数据录入等原因，造成错误的数据，即数据噪声。挖掘含噪声的数据会影响获取模式的准确性，并增加了数据挖掘的困难度。

在数据挖掘中要考虑噪声的影响，利用概率方法排除这些噪声。

(4) 数据冗余性

这表现为同一信息在多处重复出现。函数依赖是一个通常的冗余形式。冗余信息可能造成错误的数据挖掘，至少有些挖掘的知识是用户不感兴趣的。为避免这种情况的发生，数据挖掘时，需要知道数据库中有哪些固有的依赖关系。

(5) 数据稀疏性

这表现为多维数据空间中存在大量稀疏数据，稀疏数据会使数据挖掘丢失有用的模式。

(6) 海量数据

数据仓库中数据在不断增长，已出现很多海量数据仓库。数据挖掘方法需要逐步适应这种海量数据和迅速增长的数据，如建立有效的索引机制和快速查询方法等。

**2. 文本**

文本是以文字串形式表示的数据文件。文本分析包括关键词或特征提取；相似检索；文本聚类和文本分类等。

(1) 关键词或特征提取

一篇文本中，标题是该文本的高度概括。标题中的关键词是标题的核心内容。关键字的提取对于掌握该文本的内容至关重要。

文本中的特征如人名、地名、组织名等是某些文本中的主体信息，特征提取对掌握该文本的内容很重要。

(2) 相似检索

文本中的关键词的相似检索是了解文本内容的一种重要方法。例如"专家系统"与"人工智能"两个关键词是有一定联系的。研究专家系统的文本，一定属于人工智能的研究领域。

(3) 文本聚类

文本聚类是对大量文本在没有类别的情况下，利用聚类算法聚成多个类别。对于文本标题中关键词(主题字)的相似匹配是对文本聚类的一种简单方法。定义关键词的相似度，将便利文本的简单聚类，使类中的文本均满足关键词的相似度，使不同类之间的文本的关键词一定超过相似度。

(4) 文本分类

文本分类是在已经知道各文本的类别的情况下，通过分类算法获取对各已知类别的特

征描述知识(分类知识)。利用分类知识,可以对于一个新的文本区分出它属于哪个类。

### 3. 图像与视频数据

图像和视频数据是典型多媒体数据。数据以点阵信息及帧形式存储,数据量很大。图像与视频的数据挖掘包括图像与视频特征提取、基于内容的相似检索、视频镜头的编辑与组织等。

(1) 图像与视频特征提取

图像与视频数据特征有颜色、纹理和形状等。这些特征提取用于基于内容的相似检索。海水蓝色、海滩黄色、房屋的形状及颜色,需要从大量图像和视频数据中提取。

(2) 基于内容的相似检索

根据图像、视频特征的分布、比例等进行基于内容的相似检索,可以对图像和视频数据进行聚类以及分类,也能完成对新图像或视频的识别。如对遥感图像或视频的识别,这种应用非常广泛,例如森林火灾的发现与报警、河流水灾的预报等。

(3) 视频镜头的编辑与组织

镜头代表一段连续动作(视频数据流)。典型的镜头编辑如足球赛的射门、某段新闻节目等,需要在冗长的视频数据流中进行自动裁取。

经过编辑的镜头,按某种需要重新组织,将形成特定需求的新视频节目,如足球射门集锦、某个新闻事件的连续报道等。

### 4. Web 数据

随着 Internet 的发展和普及、网站数目的迅速增长以及入网人员的剧烈增多,网络数据量呈指数增长。Web 数据挖掘已成为新课题。Web 数据挖掘的特点如下。

(1) 异构数据集成和挖掘

Web 上每一个站点是一个数据源,各数据源都是异构的,形成了一个巨大的异构数据库环境。只有将这些站点的异构数据进行集成,给用户提供一个统一的视图,才能在 Web 上进行数据挖掘。

(2) 半结构化数据模型抽取

Web 上的数据非常复杂,没有特定的模型描述。虽然每个站点上的数据是结构化的,但各自的设计对整个网络是一个非完全结构化的数据,称为半结构化数据。

对半结构化数据模型的查询和集成,需要寻找一种半结构化模型抽取技术来自动抽取各站点的数据。

XML 是一种半结构化的数据模型,易于实现 Web 中信息共享与交换。

采用"实时建议"技术,能够根据用户以往的浏览行为来预测该用户以后的浏览行为,从而为用户提供个性化的浏览建议。

总之,Web 数据挖掘正在逐步成为热点。

## 6.1.3 数据挖掘任务

数据挖掘任务有六项:关联分析、时序模式、聚类、分类、偏差检测、预测。

### 1. 关联分析

关联分析是从数据库中发现知识的一类重要方法。若两个或多个数据项的取值之间重复出现且概率很高时,它就存在某种关联,可以建立起这些数据项的关联规则。

例如,买面包的顾客有90%的人还买牛奶,这是一条关联规则。若商店中将面包和牛奶放在一起销售,将会提高它们的销量。

在大型数据库中,这种关联规则是很多的,需要进行筛选,一般用“支持度”和“可信度”两个阈值来淘汰那些无用的关联规则。

“支持度”表示该规则所代表的事例(元组)占全部事例(元组)的百分比,如既买面包又买牛奶的顾客占全部顾客的百分比。

“可信度”表示该规则所代表事例占满足前提条件事例的百分比,如既买面包又买牛奶的顾客占买面包顾客中的90%,称可信度为90%。

### 2. 时序模式

通过时间序列搜索出重复发生概率较高的模式。这里强调时间序列的影响。例如,在所有购买了激光打印机的人中,半年后60%的人再购买新硒鼓,40%的人用旧硒鼓装碳粉;在所有购买了彩色电视机的人中,有60%的人再购买DVD产品。

在时序模式中,需要找出在某个最小时间内出现比率一直高于某一最小百分比(阈值)的规则。这些规则会随着形式的变化做适当的调整。

时序模式中,一个有重要影响的方法是“相似时序”。用“相似时序”的方法,要按时间顺序查看时间事件数据库,从中找出另一个或多个相似的时序事件。例如在零售市场上,找到另一个有相似销售的部门,在股市中找到有相似波动的股票。

### 3. 聚类

数据库中的数据可以划分为一系列有意义的子集,即类。简单地说,在没有类的数据中,按“距离”的远近聚集成若干类。在同一类别中,个体之间的距离较小,而不同类别上的个体之间的距离偏大。聚类增强了人们对客观现实的认识,即通过聚类建立宏观概念。例如将鸡、鸭、鹅等都聚类为家禽。

聚类方法包括统计分析方法、机器学习方法、神经网络方法等。

在统计分析方法中,聚类分析是基于距离的聚类,如欧氏距离、海明距离等。这种聚类分析方法是一种基于全局比较的聚类,它需要考察所有的个体才能决定类的划分。

在机器学习方法中,聚类是无导师的学习。在这里距离是根据概念的描述来确定的,故聚类也称概念聚类,当聚类对象动态增加时,概念聚类则称谓概念形成。

在神经网络中,自组织神经网络方法用于聚类,如ART模型、Kohonen模型等,这是一种无监督学习方法。当给定距离阈值后,各样本按阈值进行聚类。

### 4. 分类

分类是数据挖掘中应用的最多的任务。分类是在聚类的基础上,对已确定的类找出该

类别的描述知识，它代表了这类数据的整体信息，即该类的内涵描述，一般用规则或决策树模式表示。该模式能把数据库中的各元组影射到给定类别中的某一个。

一个类的内涵描述分为特征描述和辨别性描述。

特征描述是对类中对象的共同特征的描述。辨别性描述是对两个或多个类之间的区别的描述。特征描述允许不同类中具有共同特征。而辨别性描述对不同类不能有相同特征。辨别性描述用得更多。

分类是利用训练样本集（已知数据库元组和类别所组成的样本）通过有关算法而求得的。

建立分类决策树的方法，典型的有 ID3、C4.5、IBLE 等方法。建立分类规则的方法，典型的有 AQ 方法、粗糙集方法、遗传分类器等。

目前，分类方法的研究成果较多，判别方法的好坏，可从三个方面进行：①预测准确度（对非样本数据的判别准确度）；②计算复杂度（方法实现时对时间和空间的复杂度）；③模式的简洁度（在同样效果的情况下，希望决策树小或规则少）。

在数据库中，往往存在噪声数据（错误数据）、缺损值、疏密不均匀等问题。它们对分类算法获取的知识将产生坏的影响。

### 5. 偏差检测

数据库中的数据存在很多异常情况，从数据分析中发现这些异常情况也是很重要的，以便引起人们对它更多的注意。

偏差包括很多有用的知识，如：

(1) 分类中的反常实例；

(2) 模式的例外；

(3) 观察结果对模型预测的偏差；

(4) 量值随时间的变化。

偏差检测的基本方法是寻找观察结果与参照之间的差别。观察常常是某一个域的值或多个域值的汇总。参照是给定模型的预测、外界提供的标准或另一个观察。

### 6. 预测

预测是利用历史数据找出变化规律，建立模型，并用此模型来预测未来数据的种类、特征等。

典型的方法是回归分析，即利用大量的历史数据，以时间为变量建立线性或非线性回归方程。预测时，只要输入任意的时间值，通过回归方程即可求出该时间的预测值。

近年来发展起来的神经网络方法，如 BP 模型，它实现了非线性样本的学习，能进行非线性函数的判别。

分类也能进行预测，但分类一般用于离散数值；回归预测用于连续数值；神经网络方法预测既可用于连续数值，也可用于离散数值。

### 6.1.4 数据挖掘分类

数据挖掘涉及多个学科，主要包括数据库、统计学和机器学习三大主要技术。

数据库技术经过 20 世纪 80 年代的大发展，除关系数据库外，又陆续出现面向对象数据库、多媒体数据库、分布式数据库以及 Web 数据库等。数据库的应用由一般查询到模糊查询和智能查询，数据库计算已趋向并行计算。从以上各类数据库中挖掘知识正在兴起并已得到迅速发展。

统计学是一门古老的学科，现已逐渐走向社会。它已成为社会调查、了解民意以及制定决策的重要手段。

机器学习是人工智能的重要分支。它是在专家系统获取知识出现困难后发展起来的。机器学习的大部分方法和技术已演变为数据挖掘方法和技术。

数据挖掘可按数据库类型、挖掘对象、挖掘任务、挖掘方法与技术，以及应用等几方面进行分类。

#### 1. 按数据库类型分类

数据挖掘主要是在关系数据库中挖掘知识。随数据库类型的不断增加，逐步出现了不同数据库的数据挖掘，现有关系数据挖掘、模糊数据挖掘、历史数据挖掘、空间数据挖掘等多种不同数据库的数据挖掘类型。

#### 2. 按数据挖掘对象分类

数据挖掘除对数据库这个主要对象进行挖掘外，还有文本数据挖掘、多媒体数据挖掘、Web 数据挖掘。由于对象不同，挖掘的方法相差很大，文本、多媒体、Web 数据均是非结构化数据，挖掘的难度将很大。

目前，Web 数据挖掘已逐步引起人们的关注。

#### 3. 按数据挖掘任务分类

数据挖掘的任务有关联分析、时序模式、聚类、分类、偏差检测、预测等。按任务分类有关联规则挖掘、序列模式挖掘、聚类数据挖掘、分类数据挖掘、偏差分析挖掘和预测数据挖掘等类型。

各类数据挖掘由于任务不同，将会采用不同的数据挖掘方法和技术。

#### 4. 按数据挖掘方法和技术分类

数据挖掘方法和技术较多，在下一节中将详细讨论。在此对其分类进行说明。

(1) 归纳学习类

该类又分为基于信息论方法挖掘类和基于集合论方法挖掘类。基于信息论方法是在数据库中寻找信息量大的属性来建立属性的决策树。基于集合论方法是对数据库中各属性的元组集合之间关系(上、下近似关系，覆盖或排斥关系，包含关系等)来建立属性间的规则。各类中又包括多种方法，主要用于分类问题。

（2）仿生物技术类

该类又分为神经网络方法类和遗传算法类。神经网络方法是在模拟人脑神经元而建立的 MP 数学模型和 Hebb 学习规则的基础上，提出了一系列的算法模型，用于识别、预测、联想、优化、聚类等实际问题。遗传算法是模拟生物遗传过程，对选择、交叉、变异过程建立了数学算子，主要用于问题的优化和规则的生成。

（3）公式发现类

在科学实验与工程数据库中，用人工智能方法寻找和发现连续属性（变量）之间关系，建立变量之间公式，已引起人们的关注，该类中有多种数据挖掘方法，如 BACON 和 FDD 等。

（4）统计分析类

统计分析是一门独立的学科，由于能对数据库中数据求出各种不同的统计信息和知识，因此它也构成了数据挖掘中的一大类方法。

（5）模糊数学类

模糊数学是反映人们思维的一种方式。将模糊数学应用于数据挖掘各项任务中，形成了模糊数据挖掘类，如模糊聚类、模糊分类、模糊关联规则等。

（6）可视化技术类

可视化技术是一种图形显示技术。对数据的分布规律进行可视化显示或对数据挖掘过程进行可视化显示，会明显提高人们对数据挖掘的理解和挖掘效果。该技术已形成了可视化数据挖掘类的多种方法。

本书的内容将按数据挖掘的方法和技术分类的各种方法进行详细和深入的介绍，以便读者学习和使用这些方法和技术，对实际问题完成数据挖掘任务。

### 6.1.5 不完全数据处理

对不完全数据（Incomplete Data）的处理是知识发现过程中数据预处理的主要内容。在现实领域中，人们所拥有的数据常常是不完全的。在这种情况下，知识发现应该具有处理这种不完全数据并提供相应合理的近似结果的能力。

现实世界的数据库（例如商业数据库和医院数据库）中的数据很少是完全的：丢失的数据、观察不到的数据，隐藏的数据、录入过程中发生错误的数据等在现实中是经常发生的。在知识发现领域中对不完全数据的研究比较多的在于丢失的数据。

例如，在对个人调查时，被调查的对象可能会拒绝提供他的收入情况，在一项实验过程中，某些结果可能会因为某些故障而丢失，这些情况都会产生数据丢失。

关于两个变量 $X$ 和 $Y$ 的采样。其中 $X$ 是独立变量，总有观测值；$Y$ 是响应变量，可能涉及丢失值。以 $Y=?$ 代表丢失值，以（$X=i, Y=?$）代表不完全的记录。由这种简单的两个变量模型，可以推广到更一般的情况，即一个不含丢失值的变量的集合总是影响着可能具有丢失值的另一个变量。这种情况在统计学、机器学习、数据挖掘和知识发现领域里是相当常见的。

丢失数据模式分类取决于 $Y=?$ 的概率是否依赖于 $Y$ 与 $X$ 的状态。如果这一概率依赖于 $X$ 但不依赖于 $Y$，则认为数据是随机丢失的（Missing at Random）；如果 $Y=?$ 的概率既不依赖于 $Y$ 也不依赖于 $X$ 的状态，则认为数据是完全随机丢失的（Missing Completely at

Random)。对于数据随机丢失和数据完全随机丢失两种情况，如果数据挖掘方法都不受影响，那么丢失数据的模式是可以忽略的。但当 $Y=?$ 的概率既依赖于 $Y$ 又依赖于 $X$ 时，则丢失数据的模式就是不可忽略的。

处理丢失数据的方法有以下几种。

**1. 基于已知数据的方法**

忽略掉丢失的数据而只对得到的数据进行挖掘和分析。这种方法最为简单，在数据量不太大且数据是完全随机丢失的情况下可以得到令人满意的结果。但是如果数据不是随机丢失的，这种方法就不是很有效，会导致严重的偏差，这时可以采用删除有丢失数据的属性方法。

**2. 基于猜测的方法**

首先猜测被丢失的值，从而得到完全的数据，然后再运用标准的统计学和机器学习的方法进行数据挖掘和分析。具体方法有：

(1) 均值替换法：用含有丢失值的属性的已知值的平均值来代替丢失的值。

(2) 概率统计法。先求丢失值的所在属性的各取值的出现概率 $P(v_i^a)$，即表示属性 $a$ 的取值 $v_i$ 出现的概率。丢失值用出现最大概率的值 $v$ 来代替。

(3) 回归猜测。采用回归分析的方法，用未丢失的数据建立回归方程，用所依赖的变量 $X$ 求出该丢失值 $Y$。

**3. 基于模型的方法**

对于丢失值构造出一个适当的模型(非回归模型)，然后在此模型下采用恰当的方法猜测丢失的值，这是一种较为灵活的方法。

**4. 基于贝叶斯理论的方法**

利用无教师指导的贝叶斯分类技术和贝叶斯网络处理丢失的数据。

**5. 基于决策树的方法**

利用决策树和规则归纳的技术来处理丢失的数据。

以上主要讨论了对不完全数据的处理。另外，对未知的数据、隐藏的数据、错误的数据等以及这些数据和已知数据的关系，目前研究较少，还需要深入研究。

### 6.1.6 数据库的数据浓缩

数据浓缩就是在满足某种等价条件下，将复杂的难以理解的数据库，变换成简洁的、容易理解的高度浓缩的数据库。

数据浓缩包括两方面：①属性约简；②元组(记录)压缩。

**1. 属性约简**

属性约简一般用于分类问题。属性约简的原则是保持数据库中分类关系不变。目前，

属性约简一般采用粗糙集(Rough Set)方法,也可以采用信息论方法。

在数据库(S)的分类问题中,属性分为条件属性(*C*)和决策属性(*D*)。属性约简是在条件属性中删除那些不影响对决策属性进行分类的多余的属性。经过研究对条件属性一般分为可省略属性和不可省略属性。不可省略属性实质是对决策属性进行分类的核心属性(Core(S))。而可省略属性(Choice(S))并不是全部都可省略的属性,需要在可省略属性中挑选出部分属性与核心属性组合成等价原数据库的分类效果。

例如,有如下汽车数据库(CTR),有 9 个条件属性,1 个决策属性(里程),如表 6.1 所示。

**表 6.1　汽车数据库(CTR)**

| 序号 | 类型<br>a | 汽缸<br>b | 涡轮式<br>c | 燃料<br>d | 排气量<br>e | 压缩率<br>f | 功率<br>g | 换挡<br>h | 重量<br>i | 里程<br>D |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 小型 | 6 | Y | 1 型 | 中 | 高 | 高 | 自动 | 中 | 中 |
| 2 | 小型 | 6 | N | 1 型 | 中 | 中 | 高 | 手动 | 中 | 中 |
| 3 | 小型 | 6 | N | 1 型 | 中 | 高 | 高 | 手动 | 中 | 中 |
| 4 | 小型 | 4 | Y | 1 型 | 中 | 高 | 高 | 手动 | 轻 | 高 |
| 5 | 小型 | 6 | N | 1 型 | 中 | 中 | 中 | 手动 | 中 | 中 |
| 6 | 小型 | 6 | N | 2 型 | 中 | 中 | 中 | 自动 | 重 | 低 |
| 7 | 小型 | 6 | N | 1 型 | 中 | 中 | 高 | 手动 | 重 | 低 |
| 8 | 微型 | 4 | N | 2 型 | 小 | 高 | 低 | 手动 | 轻 | 高 |
| 9 | 小型 | 4 | N | 2 型 | 小 | 高 | 低 | 手动 | 中 | 中 |
| 10 | 小型 | 4 | N | 2 型 | 小 | 高 | 中 | 自动 | 中 | 中 |
| 11 | 微型 | 4 | N | 1 型 | 小 | 高 | 低 | 手动 | 轻 | 高 |
| 12 | 微型 | 4 | N | 1 型 | 中 | 中 | 中 | 手动 | 中 | 高 |
| 13 | 小型 | 4 | N | 2 型 | 中 | 中 | 中 | 手动 | 中 | 中 |
| 14 | 微型 | 4 | Y | 1 型 | 小 | 高 | 高 | 手动 | 中 | 高 |
| 15 | 微型 | 4 | N | 2 型 | 小 | 中 | 低 | 手动 | 中 | 高 |
| 16 | 小型 | 4 | Y | 1 型 | 中 | 中 | 高 | 手动 | 中 | 中 |
| 17 | 小型 | 6 | N | 1 型 | 中 | 中 | 高 | 自动 | 中 | 中 |
| 18 | 小型 | 4 | N | 1 型 | 中 | 中 | 高 | 自动 | 中 | 中 |
| 19 | 微型 | 4 | N | 1 型 | 小 | 高 | 中 | 手动 | 中 | 高 |
| 20 | 小型 | 4 | N | 1 型 | 小 | 高 | 中 | 手动 | 中 | 高 |
| 21 | 小型 | 4 | N | 2 型 | 小 | 高 | 中 | 手动 | 中 | 中 |

经过分析,可以得到:

Corse(*S*)={燃料,重量},Choice(*S*)={类型、涡轮式、汽缸、排气量、压缩率、功率、换

挡}

保持数据库(S)分类关系不变的 7 个属性约简：

(1) {类型,燃料,排气量,重量}4 个属性；

(2) {燃料,排气量,压缩率,重量}4 个属性；

(3) {类型,汽缸,燃料,压缩率,重量}5 个属性；

(4) {类型,燃料,压缩率,功率,重量}5 个属性；

(5) {类型,汽缸,燃料,功率,重量}5 个属性；

(6) {汽缸,燃料,压缩率,功率,重量}5 个属性；

(7) {类型,汽缸,涡轮式,燃料,换挡,重量}6 个属性。

以上 7 种属性约简都等价于原数据库中 9 个属性的决策分类。

其中最小属性约简是(1)和(2),用 4 个属性就可以代替数据库中 9 个属性。利用最小属性约简(2),经过进一步处理,可以得到原数据库的等价数据库,如表 6.2 所示。

**表 6.2　约简后的数据库**

| | 燃料 | 排气量 | 压缩率 | 重量 | 里程 |
|---|---|---|---|---|---|
| 1’ | * | * | * | 重 | 低 |
| 2’ | * | * | * | 轻 | 高 |
| 3’ | * | 小 | 中 | * | 高 |
| 4’ | * | 中 | * | 中 | 中 |
| 5’ | 1 型 | 小 | 高 | * | 高 |
| 6’ | 2 型 | * | 高 | 中 | 中 |

说明：“ * ”表示可不考虑该属性的取值。

### 2. 元组(记录)压缩

元组(记录)压缩实质上是对数据库的元组(记录)进行合并、归并和聚类等。

(1) 相同元组(记录)的合并

在进行属性约简后,会出现很多相同的元组,这样就可以合并这些相同的元组。

(2) 利用概念树进行归并

概念树是一种对概念的层次进行划分的树。概念树与数据库中特定的属性有关,它将各个层次的概念按从一般到特殊的顺序排列。在概念树中最一般的概念作为树的根结点;最特殊的概念作为叶结点,它对应数据库具体属性值。例如,反映某数据库中“籍贯”这个属性的概念树如图 6.2 所示。

利用概念树进行向上归纳,可以实现数据库元组归并。例如,对数据库中“籍贯”为广州、深圳、东莞、佛山等城市的所有学生的记录都归并为广东省,即放在“籍贯＝广东省”的新记录中,这样就完成了广东省内学生的多个元组(记录)都归并到一个元组(记录)中,实现了元组(记录)的压缩。对学生数据库这种元组压缩有利于学校对各省学生的生活习惯有概括的了解,便利了学校对他们的管理。

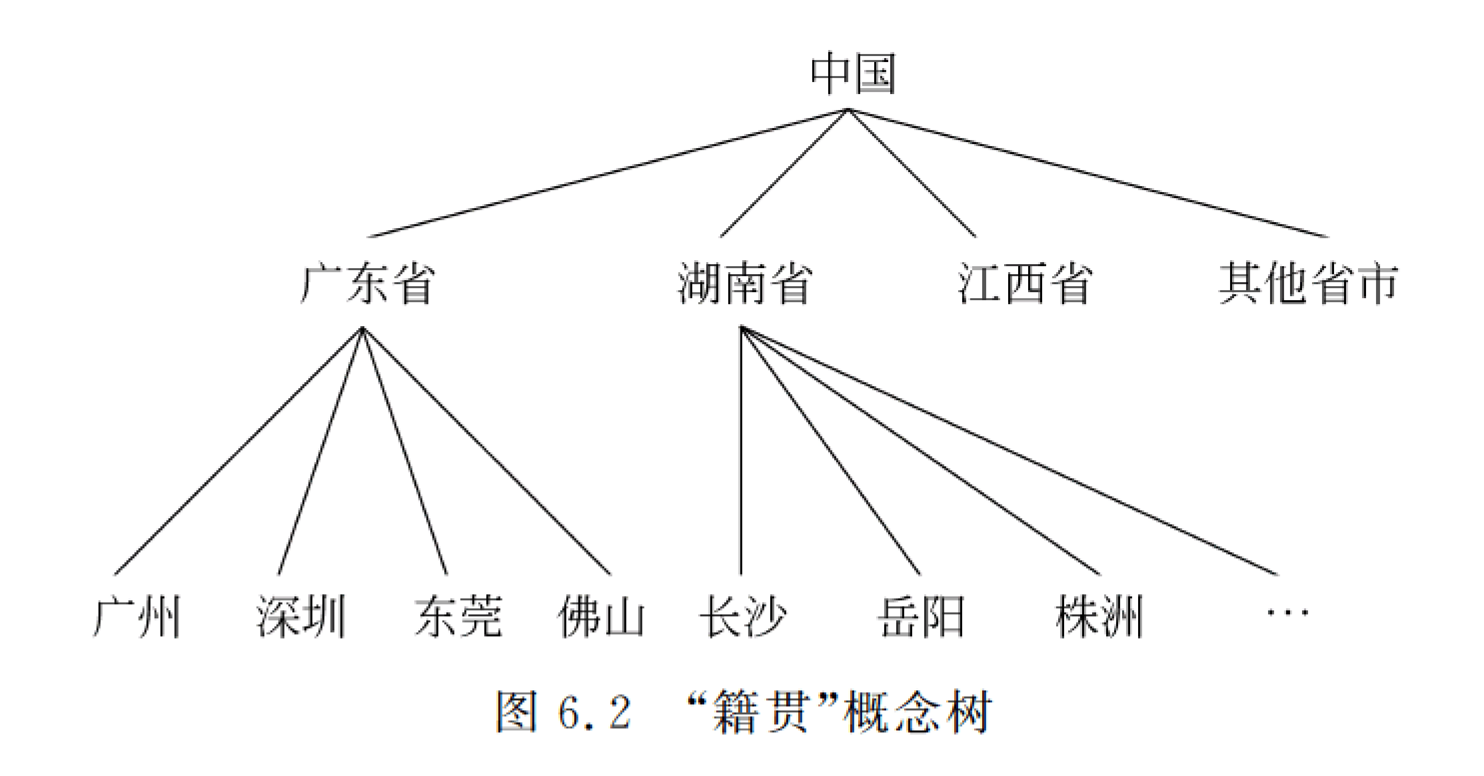



图 6.2 “籍贯”概念树

(3) 对元组的聚类

为了对数据库中所有元组(记录)有一个概括的了解,在元组之间设定一种距离方法(如海明距离),对数据库中所有元组进行聚类。这种聚类能完成对同一类的多个元组进行聚集,形成一个类元组。数据库按类元组重新组织,就完成了原数据库元组高度压缩的新数据库。

## 6.2 数据挖掘方法和技术

数据挖掘方法依据的基本原理主要有:①信息论,主要是计算数据库中属性的信息量,如 ID3、IBLE 等方法;②集合论,利用集合之间的覆盖关系(如粗糙集方法、覆盖正例排斥反例的 AQ11 方法),或计算数据项在整个集合中所占的比例(如关联规则挖掘方法);③仿生物技术,把生物体的运转过程转换成数学模型,再用数学模型去解决现实世界的非生物问题,如神经网络、遗传算法等;④人工智能技术,主要是利用启发式搜索方法,如公式发现的 BACOM、FDD 等方法;⑤可视化技术,主要是利用图形显示技术。

数据挖掘方法和技术可以分为六大类。

### 6.2.1 归纳学习的信息论方法

归纳学习方法是目前重点研究的方向,研究成果较多。从采用的技术上看,分为两大类:信息论方法(这也是常说的决策树方法)和集合论方法。每类方法又包含多个具体方法。

信息论方法是利用信息论的原理建立决策树。由于该方法最后获得的知识表示形式是决策树,因此一般文献中称它为决策树方法。该类方法的实用效果好,影响较大。

信息论方法中较有特色的方法有以下几种。

#### 1. ID3 等方法(决策树方法)

Quiulan 研制的 ID3 方法是利用信息论中互信息(Quiulan 称为信息增益)寻找数据库中具有最大信息量的字段,建立决策树的一个结点,再根据字段的不同取值建立树的分支,再由每个分支的数据子集重复建树的下层结点和分支的过程,这样就建立了决策树。这种方法对数据库愈大这种方法效果愈好。ID3 方法在国际上影响很大。ID3 方法以后又陆续

开发了 ID4、ID5、C4.5 等方法。

**2．IBLE 方法（决策规则树方法）**

钟鸣、陈文伟研制了 IBLE 方法，是利用信息论中信道容量，寻找数据库中信息量从大到小的多个字段的取值建立决策规则树的一个结点，根据该结点中指定字段取值的权值之和与两个阈值比较，建立左、中、右三个分支，在各分支子集中重复建树结点和分支的过程，这就建立了决策规则树。IBLE 方法比 ID3 方法在识别率上提高了 10 个百分点。以后又研制了 IBLE-R 方法。

## 6.2.2 归纳学习的集合论方法

集合论方法是开展较早的方法。近年来，粗糙集理论的发展使集合论方法得到了迅速的发展。这类方法中包括覆盖正例排斥反例的方法（典型的方法是 AQ 系列方法）、概念树方法和粗糙集（Rough Set）方法。关联规则挖掘方法也属于集合论方法。

**1．粗糙集（Rough Set）方法**

在数据库中将行元素看成对象，列元素是属性（分为条件属性和决策属性）。等价关系 $R$ 定义为不同对象在某个（或几个）属性上取值相同，这些满足等价关系的对象组成的集合称为该等价关系 $R$ 的等价类。条件属性上的等价类 $E$ 与决策属性上的等价类 $Y$ 之间有三种情况：①下近似：$Y$ 包含 $E$；②上近似：$Y$ 和 $E$ 的交非空；③无关：$Y$ 和 $E$ 的交为空。对下近似建立确定性规则，对上近似建立不确定性规则（含可信度），无关情况不存在规则。

**2．关联规则挖掘**

关联规则挖掘是在交易事务数据库中，挖掘出不同项（商品）集的关联关系，即发现哪些商品频繁地被顾客同时购买。

关联规则挖掘是在事务数据库 $D$ 中寻找那些不同项集（如含 $A$ 和 $B$ 两个商品）同时出现的概率（即 $P(AB)$）大于最小支持度（min_sup），且在包含一个项集（如 $A$）的所有事务中，又包含另一个项集（如 $B$）的条件概率（即 $P(B|A)$）大于最小可信度（min_conf）时，则存在关联规则（即 $A \rightarrow B$）。

**3．覆盖正例排斥反例方法**

它是利用覆盖所有正例，排斥所有反例的思想来寻找规则。比较典型的有 Michalski 的 AQ11 方法、洪家荣改进的 AQ15 方法以及洪家荣的 AE5 方法。

AQ 系列的核心算法是在正例集中任选一个种子，它到反例集中逐个比较，对字段取值构成的选择子相容则舍去，相斥则保留。按此思想循环所有正例种子，将得到正例集的规则（选择子的合取式）。

AE 系列方法是在扩张矩阵中寻找覆盖正例排斥反例的字段值的公共路（规则）。

#### 4. 概念树方法

数据库中记录的属性字段按归类方式进行合并，建立起来的层次结构称为概念树。例如对“城市”概念树的最下层是具体市名或县名（如长沙、南京等），它的直接上层是省名（湖南、江苏等），省名的直接上层是国家行政区（华南、华东等），再上层是国名（中国、日本等）。

利用概念树提升的方法可以大大浓缩数据库中的记录（元组）。对多个属性字段的概念树提升，将得到高度概括的知识基表，然后再将它转换成规则。

### 6.2.3 仿生物技术的神经网络方法

仿生物技术典型的方法是神经网络方法和遗传算法。这两类方法已经形成了独立的研究体系。它们在数据挖掘中也发挥了巨大的作用，可以将它们归并为仿生物技术类。

神经网络方法模拟了人脑神经元结构，是以 MP 数学模型和 Hebb 学习规则为基础的，建立了三大类多种神经网络模型。

#### 1. 前馈式网络

它以感知机、BP 反向传播模型、函数型网络为代表。此类网络可用于预测、模式识别等方面。

#### 2. 反馈式网络

它以 Hopfield 的离散模型和连续模型为代表，分别用于联想记忆和优化计算。

#### 3. 自组织网络

它以 ART 模型、Kohonen 模型为代表，用于聚类。

神经网络的知识体现在网络连接的权值上，是一个分布式矩阵结构。神经网络的学习体现在神经网络权值的逐步计算上（包括反复迭代或累加计算）。

### 6.2.4 仿生物技术的遗传算法

这是模拟生物进化过程的算法。它由三个基本算子组成：

#### 1. 繁殖（选择）

从一个旧种群（父代）选择出生命力强的个体产生新种群（后代）的过程。

#### 2. 交叉（重组）

选择两个不同个体（染色体）的部分（基因）进行交换，形成两个新个体。

#### 3. 变异（突变）

对某些个体的某些基因进行变异（1 变 0，0 变 1），形成新个体。

这种遗传算法起到产生优良后代的作用。这些后代需要满足适应值，经过若干代的遗

传，将得到满足要求的后代（问题的解）。遗传算法已在优化计算和分类机器学习方面发挥了显著的效果。

### 6.2.5 数值数据的公式发现

在工程和科学数据库（由实验数据组成）中，利用人工智能启发式搜索方法（反复试验），对若干数据项（变量）进行一定的数学运算，可求得相应的数学公式。

#### 1. 物理定律发现系统 BACON

BACON 发现系统完成了物理学中大量定律的重新发现。它的基本思想是对数据项反复进行初等数学运算（加、减、乘、除等）形成的组合数据项，若它的值为常数（启发式），就得到了组合数据项等于常数的公式。该系统有 5 个版本，分别为 BACON. 1 到 BACON. 5。

#### 2. 经验公式发现系统 FDD

陈文伟等人研制了 FDD 发现系统。基本思想是对两个数据项交替取初等函数后与另一数据项的线性组合，反复进行不同的初等函数试验，当线性组合为直线时（启发式），就找到了数据项（变量）的初等函数的线性组合公式。该系统所发现的公式比 BACON 系统发现的公式更宽些。该系统有 3 个版本，分别为 FDD. 1 到 FDD. 3。

### 6.2.6 可视化技术

可视化技术是一种图形显示技术。例如，把数据库中多维数据变成多种图形，这对于揭示数据中内在本质以及分布规律起到很强的作用。对数据挖掘过程可视化，并进行人机交互可提高数据挖掘的效果。

数据可视化是创建二维或三维业务数据集的图表，使得用户用于理解业务数据，从而提升知识和洞察力。例如，多维数据的多维结构类型（MTS）图与多维表格是对多维数据可视化的显示。利用直方图（二维）、柱形图（三维）、饼图、折线图、雷达图、散点图等能更形象地表示数据之间对比与变化的关系。

可视化数据挖掘是创建可视化的数据挖掘模型，利用这些模型发现业务数据集中存在的模式，从而辅助决策支持及预测新的商机。

可视化技术的基本工作如下。

#### 1. 提取几何图元

这是可视化系统的主要部分，由不同类型的数据（点、线）构造成表面或体素模型。它是构造、仿真、分析数据分布模型的有效手段。

#### 2. 绘制

这是利用计算机图形学中的成果，包括图像生成、消隐、光照效应及绘制等步骤。

### 3. 显示和演放

为了取得有效的显示效果，这一步骤将提供图片组合、文件标准、着色、旋转、放大、存储等功能。

可视化绘制(render)方法就是把隐藏于大容量数据集中的物理信息转化为有组织结构表示的视觉信号集合，如空间几何形状、颜色、亮度等。目前常用的可视化绘制方法有几何法、彩色法、多媒体法和光学法。

# 6.3 数据挖掘的知识表示

数据挖掘各种方法获得的知识的表示形式，主要有六种：规则、决策树、知识基(浓缩数据)、网络权值、公式和案例。

## 6.3.1 规则知识

规则知识由前提条件和结论两部分组成。前提条件由字段项(属性)的取值的合取(与∧)和析取(或∨)组合而成，结论为决策字段项(属性)的取值或者类别组成。

下面用一个简单例子进行说明，如两类人数据库的 9 个元组(记录)如表 6.3 所示。

表 6.3 两类人数据库

| | 身高 | 头发 | 眼睛 | | 身高 | 头发 | 眼睛 |
|---|---|---|---|---|---|---|---|
| 第一类人 | 矮 | 金色 | 蓝色 | 第二类人 | 高 | 金色 | 黑色 |
| | 高 | 红色 | 蓝色 | | 矮 | 黑色 | 蓝色 |
| | 高 | 金色 | 蓝色 | | 高 | 黑色 | 蓝色 |
| | 矮 | 金色 | 灰色 | | 高 | 黑色 | 灰色 |
| | | | | | 矮 | 金色 | 黑色 |

利用上面介绍的数据挖掘方法，将能很快得到如下规则知识：

IF(发色=金色∨红色)∧(眼睛=蓝色∨灰色)THEN 第一类人

IF(发色=黑色)∨(眼睛=黑色)THEN 第二类人

即凡是具有金色或红色的头发，并且同时具有蓝色或灰色眼睛的人属于第一类人；凡是具有黑色头发或黑色眼睛的人属于第二类人。

## 6.3.2 决策树知识

数据挖掘的信息论方法所获得的知识一般表示为决策树。

如 ID3 方法的决策树是由信息量最大的字段(属性)作为根结点，它的各个取值为分支，对各个分支所划分的数据元组(记录)子集，重复建树过程，扩展决策树，最后得到相同类别的子集，以该类别作为叶结点。

例如：上例的两类人数据库，按 ID3 方法得到的决策树如图 6.3 所示。

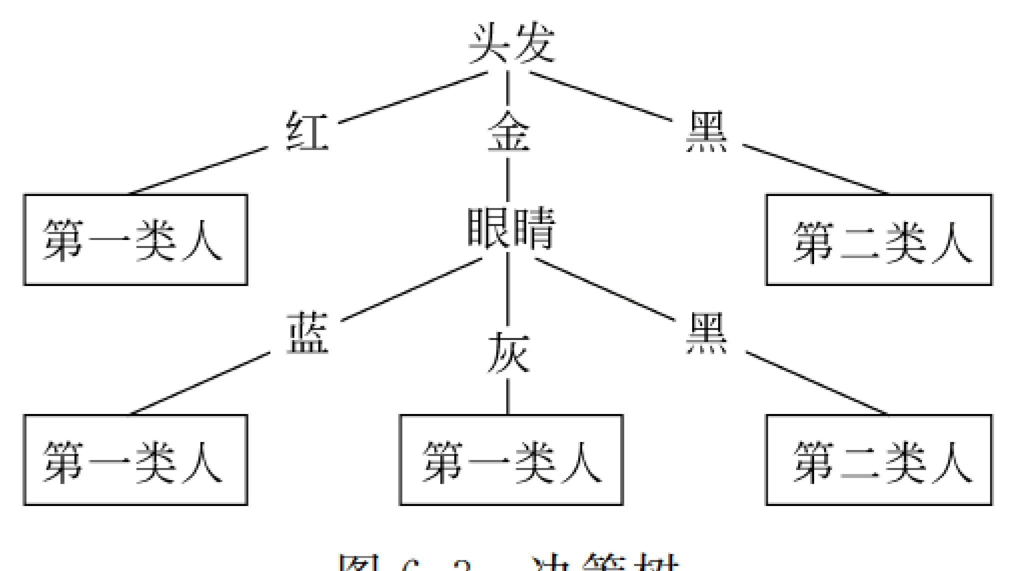



图 6.3　决策树

### 6.3.3　知识基（浓缩数据）

在知识发现过程的数据准备中，数据转换的一项属性约简工作就是找出可省略的属性。在删除不必要的属性后，对数据库中出现的相同的元组（记录）进行合并。这样，通过属性约简方法能压缩数据库的属性和相应的元组，最后得到浓缩数据，称为知识基。它是原数据库的精华，很容易转换成规则知识。

例如上例中两类人的数据库，通过属性约简计算可以得出身高是不必要的属性，删除它后，再合并相同数据元组，得到浓缩数据如表 6.4 所示。

**表 6.4　知识基（浓缩数据）**

| | 头发 | 眼睛 | | 头发 | 眼睛 |
|---|---|---|---|---|---|
| 1 类人 | 金色 | 蓝色 | 2 类人 | 金色 | 黑色 |
| 1 类人 | 红色 | 蓝色 | 2 类人 | 黑色 | 蓝色 |
| 1 类人 | 金色 | 灰色 | 2 类人 | 黑色 | 灰色 |

### 6.3.4　神经网络权值

神经网络方法经过对训练样本的学习后，所得到的知识是网络连接权值和结点的阈值，一般表示为矩阵和向量。例如，异或问题的网络权值和阈值如图 6.4 所示。

输入层网络权值：

$$\begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

隐结点阈值：

$$\begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} = \begin{pmatrix} 0.5 \\ 1.5 \end{pmatrix}$$

输入层网络权值：

$$(T_1, T_2) = (-1, 1)$$

输出结点阈值：

$\phi=0.$

$\phi$　$Z$　$T_1$　$T_2$　$\theta_1$　$y_1$　$\theta_2$　$y_2$　$w_{12}$　$w_{21}$　$w_{11}$　$w_{22}$　$X_1$　$X_2$

图 6.4　神经网络结构和权值

### 6.3.5 公式知识

对于科学和工程数据库，一般存放的是大量实验数据(数值)。它们中蕴涵着一定的规律性，通过公式发现算法，可以找出各种变量间的相互关系，并用公式表示。

例如，太阳系行星运动数据中包含行星运动周期(旋转一周所需时间，天)，以及它与太阳的距离(围绕太阳旋转的椭圆轨道的长半轴，百万公里)，数据如表 6.5 所示。

**表 6.5 太阳系行星数据**

| | 水星 | 金星 | 地球 | 火星 | 木星 | 土星 |
|---|---|---|---|---|---|---|
| 周期 $P$ | 88 | 225 | 365 | 687 | 4343.5 | 10767.5 |
| 距离 $d$ | 58 | 108 | 149 | 228 | 778 | 1430 |

通过物理定律发现系统 BACON 和经验公式发现系统 FDD 均可以得到开普勒第三定律：

$$d^3/p^2 = 25$$

### 6.3.6 案例

案例是人们经历过的一次完整的事件。当人们为解决一个新问题时，总是先回顾自己以前处理过的类似事件(案例)。将以前案例中解决问题的方法或者处理的结果作为参考并进行适当的修改，以解决当前新问题。利用这种思想建立起基于案例推理(CBR，Case Based Reasoning)。CBR 的基础是案例库，在案例库中存放大量成功或失败的案例。CBR 利用相似检索技术，对新问题到案例库中搜索相似案例，再经过对旧案例的修改来解决新问题。

可见，案例是解决新问题的一种知识。案例知识一般表示为三元组：

<问题描述，解描述，效果描述>

- 问题描述：对求解问题及周围世界或环境的所有特征的描述；
- 解描述：对问题求解方案的描述；
- 效果描述：描述解决方案后的结果情况，是失败还是成功。

## 习 题 6

1. 数据挖掘与知识发现两个概念有什么不同？
2. 知识发现过程由哪三部分组成？每部分的工作是什么？
3. 数据挖掘的对象有哪些？它们各自的特点是什么？
4. 数据挖掘的任务有哪些？每项任务的含义是什么？
5. 聚类与分类有什么不同？
6. 如何产生不完全数据？
7. 数据是随机丢失的概念是什么？

8. 数据是完全随机丢失的概念是什么?
9. 哪种丢失数据的模式是可以忽略的?
10. 哪种丢失数据的模式是不可以忽略的?
11. 处理丢失数据的方法有哪些?
12. 数据浓缩包括哪两个方面?
13. 属性约简的原则是什么?
14. 属性约简一般采用哪些方法?
15. 元组压缩有哪几种?
16. 如何利用概念树进行元组的压缩?
17. ID3 方法建立决策树的基本思想是什么?
18. “信息增益”是“互信息”吗?
19. 粗糙集方法如何获得规则?
20. 神经网络方法有哪几类?
21. 遗传算法的三个算子是什么?
22. 公式发现中的 BACON 方法与 FDD 方法的基本思想是什么?
23. 数据挖掘的知识表示有哪些?
24. 规则知识与决策树知识和知识基是等价的吗?
25. 人类社会的知识表示是什么? 它与计算机中的知识表示有什么不同?
26. 为什么要研究计算机中的知识表示?

# 第7章 信息论方法

信息论原理是数据挖掘的理论基础之一。一般用于分类问题，即从大量数据中获取分类知识。具体来说，就是在已知各实例的类别的数据中，找出确定类别的关键的条件属性。求关键属性的方法，即先计算各条件属性的信息量，再从中选出信息量最大的属性，信息量的计算是利用信息论原理中的公式。获取的分类知识表示形式为：

(1) 决策树，如 ID3、C4.5 方法，是把信息量最大的属性作为树或子树的根结点，属性的取值作为分支。

(2) 决策规则树，如 IBLE 方法，是把信息量大的多个属性作为树或子树的结点，多个属性的权值和与阈值比较大小来产生分支。

## 7.1 信息论原理

信息论是 C. E. Shannon 为解决信息传递(通信)过程问题而建立的理论，也称为统计通信理论。一个传递信息的系统是由发送端(信源)和接收端(信宿)以及连接两者的通道(信道)三者组成。信息论把通信过程看做是在随机干扰的环境中传递信息的过程。在这个通信模型中，信息源和干扰(噪声)都被理解为某种随机过程或随机序列。因此，在进行实际的通信之前，收信者(信宿)不可能确切了解信源究竟会发出什么样的具体信息，不可能判断信源会处于什么样的状态。这种情形就称为信宿对于信源状态具有不确定性。而且这种不确定性是存在于通信之前的，因而又叫做先验不确定性。

在进行了通信之后，信宿收到了信源发来的信息，这种先验不确定性才会被消除或者被减少。如果干扰很小，不会对传递的信息产生任何可察觉的影响，信源发出的信息能够被信宿全部收到，在这种情况下，信宿的先验不确定性就会被完全消除。但是，在一般情况下，干扰总会对信源发出的信息造成某种破坏，使信宿收到的信息不完全。因此，先验不确定性不能全部被消除，只能部分地消除。换句话说，通信结束之后，信宿还仍然具有一定程度的不确定性。这就是后验不确定性。显然，后验不确定性总要小于先验不确定性，不可能大于先验不确定性。

(1) 如果后验不确定性的大小正好等于先验不确定性的大小，这就表示信宿根本没有收到信息。

(2) 如果后验不确定性的大小等于零，这就表示信宿收到了全部信息。

可见，信息是用来消除(随机)不确定性的度量。信息量的大小，由所消除的不确定性的大小来计量。

### 7.1.1 信道模型和学习信道模型

#### 1. 信道模型

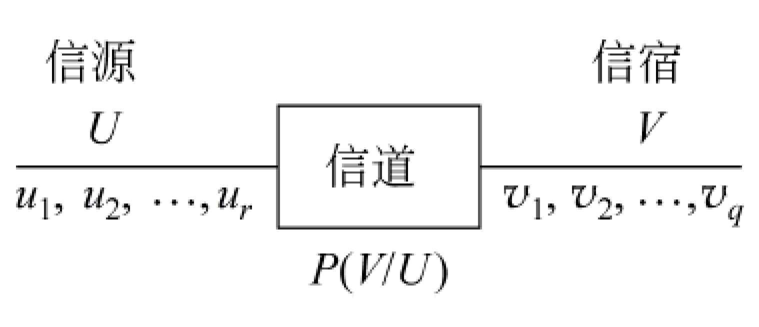



图 7.1 信道模型

信息论的信道模型如图 7.1 所示。信源发出的符号 U 取值为 $u_1, u_2, \cdots, u_r$，信宿接收的符号 $V$ 取值为 $v_1, v_2, \cdots, v_q$。

条件概率 $P(V|U)$，称为信道的传输概率或转移概率，它反映信道的输入与输出的关系，用矩阵来表示称为转移概率矩阵。

$$\begin{bmatrix} P(v_1/u_1) & P(v_2/u_1) & \cdots & P(v_q/u_1) \\ P(v_1/u_2) & P(v_2/u_2) & \cdots & P(v_q/u_2) \\ \vdots & \vdots & \vdots & \vdots \\ P(v_1/u_r) & P(v_2/u_r) & \cdots & P(v_q/u_r) \end{bmatrix} \tag{7.1}$$

其中，$\sum_{j=1}^{q} P(v_j/u_i) = 1$，$i = 1, 2, \cdots, r$。

转移概率 $P(v_j/u_i)$表示收到信息 $v_j$ 后判定输入为 $u_i$ 的概率。

信道的数学模型可用三元组$(U, P(V|U), V)$来表示，给定三元组后信道就给定了。给定了信道，将要研究在信宿收到符号 $V$ 的值 $v_j$ 后，如何正确判定信源发出的符号 $U$ 的是哪个值 $u_i$？

#### 2. 学习信道模型

学习信道模型是信息模型应用于机器学习和数据挖掘的具体化。学习信道模型的信源是实体的类别，简单采用“是”、“非”两类，令实体类别 $U$ 的值域为$\{u_1, u_2\}$，$U$ 取 $u_1$ 表示取“是”类中任一例子，取 $u_2$ 表示取“非”类中任一例子。信宿是实体的特征(属性)取值。实体中某个特征(属性)$V$，它的值域为$\{v_1, v_2, \cdots, v_q\}$。

U　学习信道　V

$u_1, u_2$　$P(V/U)$　$v_1, v_2, \cdots, v_q$

图 7.2 学习信道模型

一般把实体中的类别 $U$ 看成输入，把某特征的取值 $V$ 看成输出，建立“学习信道模型”，如图 7.2 所示。

建立学习信道模型后，就可以利用信息论的信道模型原理来解决归纳学习和数据挖掘的问题。

### 7.1.2 信息熵与条件熵

#### 1. 信源数学模型

消息(符号)$u_i(i=1,2,\cdots,r)$的发生概率 $P(u_i)$组成信源数学模型(样本空间和概率空间)

$$[U, P] = \begin{bmatrix} u_1 & u_2 & \cdots & u_r \\ P(u_1) & P(u_2) & \cdots & P(u_r) \end{bmatrix} \tag{7.2}$$

#### 2. 自信息

单个消息 $u_i$ 发出前的不确定性(随机性)称为自信息，定义为

$$I(u_i) = \log \frac{1}{P(u_i)} = -\log P(u_i) \tag{7.3}$$

以 2 为底，所得的信息量单位为 $b$。

**3. 信息熵**

信息熵是自信息的平均值(数学期望)。它反映了信源 $U$ 中所有消息在发出前的平均不确定性。定义为

$$H(U) = \sum_i P(u_i)\log \frac{1}{P(u_i)} = -\sum_i P(u_i)\log P(u_i) \tag{7.4}$$

信息熵 $H(U)$是信源 $U$ 发出前的平均不确定性，也称先验熵。

$H(U)$的性质：

(1) $H(U)=0$ 时，说明只存在着唯一的可能性，不存在不确定性。

(2) 如果 $n$ 种可能的发生都有相同的概率，即所有的 $u_i$ 有 $P(u_i)=1/n$，$H(U)$达到最大值 $\log n$，系统的不确定性最大。

(3) $P(u_i)$互相接近，$H(U)$就大。$P(u_i)$相差大，则 $H(U)$就小。

如果信道中无干扰(噪声)，信道输出符号 $v_j$ 与输入符号 $u_i$ 一一对应，那么接收到传送过来的符号后就消除了对发送符号的先验不确定性。

**4. 后验熵**

信宿当接收到单个输出符号 $V=v_j$后，对应输入符号 $U=u_i$ 的后验概率是 $P(u_i|v_j)$，对应输入符号集 $U$ 的后验概率分布是 $P(U|v_j)$。那么接收到单个输出符号 $V=v_j$ 后，关于 $U$ 的平均不确定性为

$$H(U \mid v_j) = \sum_i P(u_i \mid v_j)\log \frac{1}{P(u_i \mid v_j)} \tag{7.5}$$

这是接收到单个输出符号 $v_j$ 后关于信源集 $U$ 的不确定性，称为后验熵。后验熵是当信道接收端接收到单个输出符号 $v_j$ 后，关于输入符号 $U$ 的信息度量。

**5. 条件熵**

后验熵对输出符号集 $V$(所有符号)求平均值(数学期望)，得到条件熵：

$$H(U \mid V) = \sum_j P(v_j)\sum_i P(u_i \mid v_j)\log \frac{1}{P(u_i \mid v_j)} \tag{7.6}$$

条件熵表示在输出端收到全部符号 $V$ 后，对于输入端的全部符号集 $U$ 尚存在的不确定性(信道疑义度)。对 $U$ 集存在的不确定性是由于干扰(噪声)引起的。如果是一一对应信道，那么接收到符号集 $V$ 后，对 $U$ 集的不确定性完全消除，则信道疑义度 $H(U|V)=0$。

从上面的分析可知：条件熵小于无条件熵，即 $H(U|V)<H(U)$。说明接收到符号集 $V$ 的所有符号后，关于输入符号 $U$ 的平均不确定性减少了，即总能消除一些关于输入端 $U$ 的不确定性，从而获得了一些信息。

### 7.1.3 互信息与信息增益

$H(U)$代表接收到输出符号集 $V$ 以前关于输入符号集 $U$ 的平均不确定性，而 $H(U|V)$

代表收到输出符号集 $V$ 后关于输入符号 $U$ 的平均不确定性。可见，通过信道传输消除了一些不确定性，获得了一定的信息。定义为

$$I(U,V) = H(U) - H(U \mid V) \tag{7.7}$$

$I(U,V)$称为 $U$ 和 $V$ 之间的互信息，它代表接收到符号集 $V$ 后获得的关于 $U$ 的信息量。

可见，熵($H(U)$、$H(U|V)$)只是平均不确定性的描述。熵差($H(U)-H(U|V)$)是不确定性的消除，即互信息才是接收端所获得的信息量。

对于学习信道模型的输入端 $U$ 只有 $u_1$，$u_2$ 两类，互信息的计算公式为

$$H(U) = \sum_{i}^{2} P(u_i) \log \frac{1}{P(u_i)} \tag{7.8}$$

$$H(U \mid V) = \sum_{j} P(v_j) \sum_{i} P(u_i \mid v_j) \log \frac{1}{P(u_i \mid v_j)} \tag{7.9}$$

$$I(U,V) = H(U) - H(U \mid V) \tag{7.10}$$

当 $P(u_i)$或 $P(u_i|v_j)$为零时，定义对数为零。

J. R. Quinlan 在提出 ID3 方法时，用“信息增益”概念，实际上是信息论中的“互信息”概念。

### 7.1.4 信道容量与译码准则

#### 1. 信道容量

给定信道的互信息 $I(U,V)$是 $P(U)$的∩型函数。由∩型函数的性质知道，一定存在一概率分布 $P(U)$，使得 $I(U,V)$达到最大。这个最大的互信息就称为信道容量(Capacity)，记为 $C$。

$$C = \underset{P(U)}{\mathrm{Max}} \{I(U,V)\} \tag{7.11}$$

无论 $P(U)$如何变化，$I(U,V)$总不会大于 $C$。因此 $C$ 对给定信道是个常数。

若以 $C$ 作为特征选择量，去掉 $C$ 小的特征(信息量小的特征)，选择 $C$ 大的特征(信息量大的特征)，即 $C$ 大的特征对区分正反例更有效。

互信息 $I(U,V)$的计算会随实例个数的变化而变化，而信道容量 $C$ 不会随实体个数的多少而变化，用 $C$ 作为特征的信息量更准确。但是，$C$ 的计算极为复杂，一般要用计算机做迭代运算。

#### 2. 译码准则

信息论方法需要选择信道，然后根据输出判定输入是什么类别。

这里只研究二元信道译码准则，多元信道可以转换为二元信道。二元信道如图 7.3 所示。



图 7.3　二元信道

将其中转移概率用矩阵表示为：$\begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix}$。

举一个简单的例子，设有二元信道，其转移概率矩阵为 $\boldsymbol{p} = \begin{bmatrix} 1/3 & 2/3 \\ 2/3 & 1/3 \end{bmatrix}$。

当得到特征值 $v_1$ 时，若判定实体的类别为 $u_1$，则译对的可能性 $P_{11}$ 为 1/3，译错的可能性 $P_{21}$ 为 2/3。反之得到 $v_1$ 时译成 $u_2$，则译对的可能性 $P_{21}$ 为 2/3，译错的可能性 $P_{11}$ 为 1/3。可见译错的概率既与信道的统计特性有关又与译码准则有关。

现在要定义一个译码准则。设信道如图 7.3 所示，定义译码准则就是要设计一个函数 $F(v_j)$ 对于输出的每一个 $v_j$ 唯一确定输入的一个类别 $u_i$ 与之对应（单值函数）。

二元信道，可以定义译码准则：

$$A:\begin{bmatrix} F(v_1)=u_1 \\ F(v_2)=u_2 \end{bmatrix} \quad \text{或者} \quad B:\begin{bmatrix} F(v_1)=u_2 \\ F(v_2)=u_1 \end{bmatrix}$$

还可以有另外的定义方法。

使平均错误概率最小的译码规则是最大后验概率准则，即：

后验概率 $P(u_i/v_j)$ 表示输入 $u_i$ 发生以后，$v_j$ 出现的概率。用 $P(u_*/v_j)$ 表示 $P(u_1/v_j)$ 与 $P(u_2/v_j)$ 中某一个。

当满足条件 $P(u_*|v_j) \geqslant P(u_i|v_j)$，$i=1,2$ 时定义译码函数 $F(v_j)=u_*$。

其中 $u_*$ 是 $u_1$ 和 $u_2$ 中的某一个。可以证明该准则的平均错误概率最小，即把每个 $v_j$ 判成具有最大后验概率 $P(u_i|v_j)$ 的那个类别。这个准则称为“最大后验概率准则”或“最小错误概率准则”。

## 7.2 决策树方法

### 7.2.1 决策树概念

决策树是用样本的属性作为结点，用属性的取值作为分支的树结构。它是利用信息论原理对大量样本的属性进行分析和归纳而产生的。决策树的根结点是所有样本中信息量最大的属性。树的中间结点是该结点为根的子树所包含的样本子集中信息量最大的属性。决策树的叶结点是样本的类别值。

决策树用于对新样本的分类，即通过决策树对新样本属性值的测试，从树的根结点开始，按照样本属性的取值，逐渐沿着决策树向下，直到树的叶结点，该叶结点表示的类别就是新样本的类别。决策树方法是数据挖掘中非常有效的分类方法。

决策树是一种知识表示形式，它是对所有样本数据的高度概括，即决策树能准确地识别所有样本的类别，也能有效地识别新样本的类别。

决策树概念最早出现在 CLS（Concept Learning System）中，影响最大的是 J. R. Quinlan 于 1986 年提出的 ID3 方法，他提出用信息增益（即信息论中的互信息）来选择属性作为决策树的结点。由于决策树的建树算法思想简单，识别样本效率高的特点，使 ID3 方法成为当时机器学习领域中最有影响的方法之一。后来，不少学者提出了改进 ID3 的方法，比较有影响的是 ID4、ID5 方法。J. R. Quinlan 本人于 1993 年提出了改进 ID3 的 C4.5 方法，C4.5 方法是用信息增益率来选择属性作为决策树的结点，这样建立的决策树识别样本的效率更提高了。C4.5 方法还增加剪枝、连续属性的离散化、产生规则等功能。它使决策树方法再一次得到了提高。从 ID3 方法到 C4.5 方法，决策树的结点均由单个属性构成，缺少不

同属性之间的关系。

本书作者领导的课题组在研究信息论以后，于1991年提出了基于信道容量的IBLE方法和1994年提出的基于归一化互信息的IBLE-R方法。此两方法建立的是决策规则树。树的结点是由多个属性组成的。这样，在树的结点中体现了多个属性的相互关系。由于信道容量是互信息的最大值，它不随样本数的改变而改变，从而使IBLE方法在样本识别效率上，比ID3方法提高了10个百分点。IBLE-R方法在IBLE方法的基础上增加了产生规则的功能。

决策树方法ID3和C4.5以及决策规则树方法IBLE和IBLE-R的理论基础都是信息论。

### 7.2.2 ID3方法基本思想

J. R. Quinlan的ID3方法，它的前身是CLS方法。Hunt提出的CLS的工作过程为：首先找出有判别力的属性，把数据分成多个子集，每个子集又选择有判别力的属性进行划分，一直进行到所有子集仅包含同一类型的数据为止。最后得到一棵决策树，可以用它来对新的样例进行分类。CLS的不足是没有说明如何选择有判断力的属性。

J. R. Quinlan的工作主要是引进了信息论中的互信息，他将其称为信息增益(information gain)，作为特征(属性)判别能力的度量，并且将建树的方法嵌在一个迭代的外壳之中。

在一实体世界中，每个实体用多个特征来描述。每个特征限于在一个离散集中取互斥的值。例如，设实体是某天早晨，分类任务是关于气候的类型，特征(属性)为：

(1) 天气。取值为：晴，多云，雨。

(2) 气温。取值为：冷，适中，热。

(3) 湿度。取值为：高，正常。

(4) 风。取值为：有风，无风。

每个实体属于不同的类别，为简单起见，假定仅有两个类别，分别为$P$，$N$。在这种两个类别的归纳任务中，$P$类和$N$类的实体分别称为概念的正例和反例。将一些已知的正例和反例放在一起便得到训练集。

表7.1给出一个训练集。由归纳学习算法ID3算法得出一棵正确分类训练集中每个实体的决策树，如图7.4所示。该决策树能对训练集中的每个实体，按特征取值，判别出它属于$P$、$N$中的一类。

**表7.1 气候训练集**

| 序号 | 属性 | | | | 类别 |
|---|---|---|---|---|---|
| | 天气 | 气温 | 湿度 | 风 | |
| 1 | 晴 | 热 | 高 | 无风 | $N$ |
| 2 | 晴 | 热 | 高 | 有风 | $N$ |
| 3 | 多云 | 热 | 高 | 无风 | $P$ |
| 4 | 雨 | 适中 | 高 | 无风 | $P$ |
| 5 | 雨 | 冷 | 正常 | 无风 | $P$ |

续表

| 序号 | 属性 | | | | 类别 |
| --- | --- | --- | --- | --- | --- |
| | 天气 | 气温 | 湿度 | 风 | |
| 6 | 雨 | 冷 | 正常 | 有风 | $N$ |
| 7 | 多云 | 冷 | 正常 | 有风 | $P$ |
| 8 | 晴 | 适中 | 高 | 无风 | $N$ |
| 9 | 晴 | 冷 | 正常 | 无风 | $P$ |
| 10 | 雨 | 适中 | 正常 | 无风 | $P$ |
| 11 | 晴 | 适中 | 正常 | 有风 | $P$ |
| 12 | 多云 | 适中 | 高 | 有风 | $P$ |
| 13 | 多云 | 热 | 正常 | 无风 | $P$ |
| 14 | 雨 | 适中 | 高 | 有风 | $N$ |

决策树叶子为类别名，即 $P$ 或者 $N$。其他结点由实体的特征组成，每个特征的不同取值对应一个分支。若要对一个新实体进行分类，需要从树根开始进行测试，按特征的取值分支向下进入下层结点，对该结点进行测试，过程一直进行到叶结点，实体被判为属于该叶结点所标记的类别。现有训练集外的一个例子，某天早晨气候描述为：①天气：多云；②气温：冷；③湿度：正常；④风：无风。它属于哪类气候呢？用图 7.4 来判别，可以得该实体的类别为 $P$ 类。

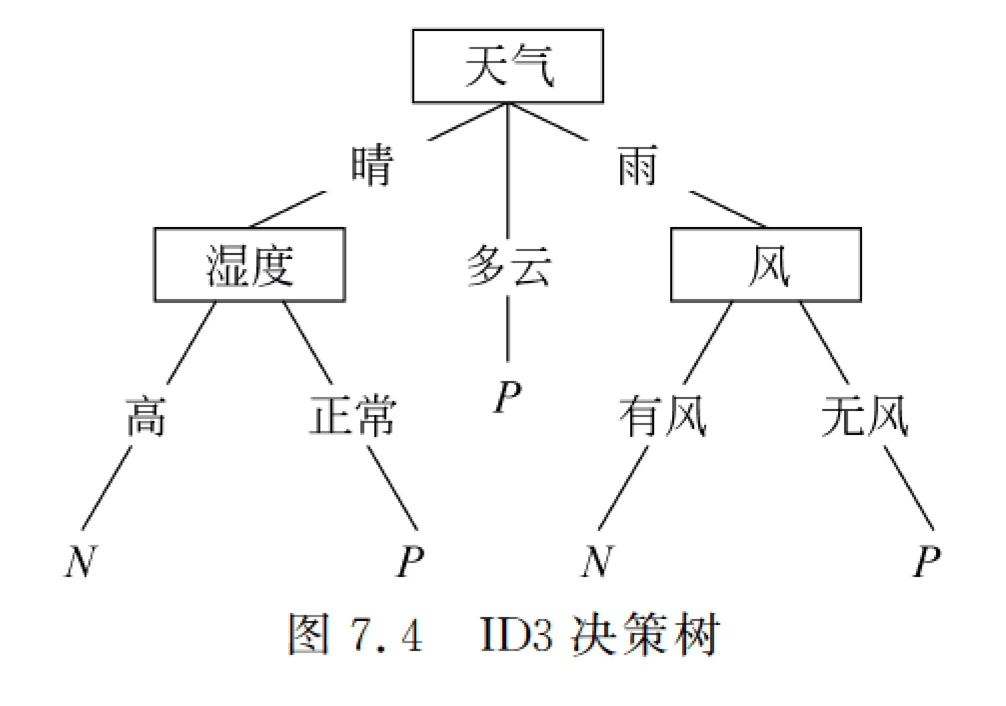


图 7.4 ID3 决策树

实际上，能正确分类训练集的决策树不止一棵。Quinlan 的 ID3 算法能得出结点最少的决策树。

## 7.2.3 ID3 算法

### 1. 主算法

(1) 从训练集中随机选择一个既含正例又含反例的子集(称为“窗口”)；

(2) 用“建树算法”对当前窗口形成一棵决策树；

(3) 对训练集(窗口除外)中例子用所得决策树进行类别判定，找出错判的例子；

(4) 若存在错判的例子，把它们插入窗口，转 2，否则结束。

主算法流程用图 7.5 表示。其中 PE、NE 分别表示正例集和反例集，它们共同组成训练集。PE'、PE"和 NE',NE"分别表示正例集和反例集的子集。

主算法中每迭代循环一次，生成的决策树将会不相同。

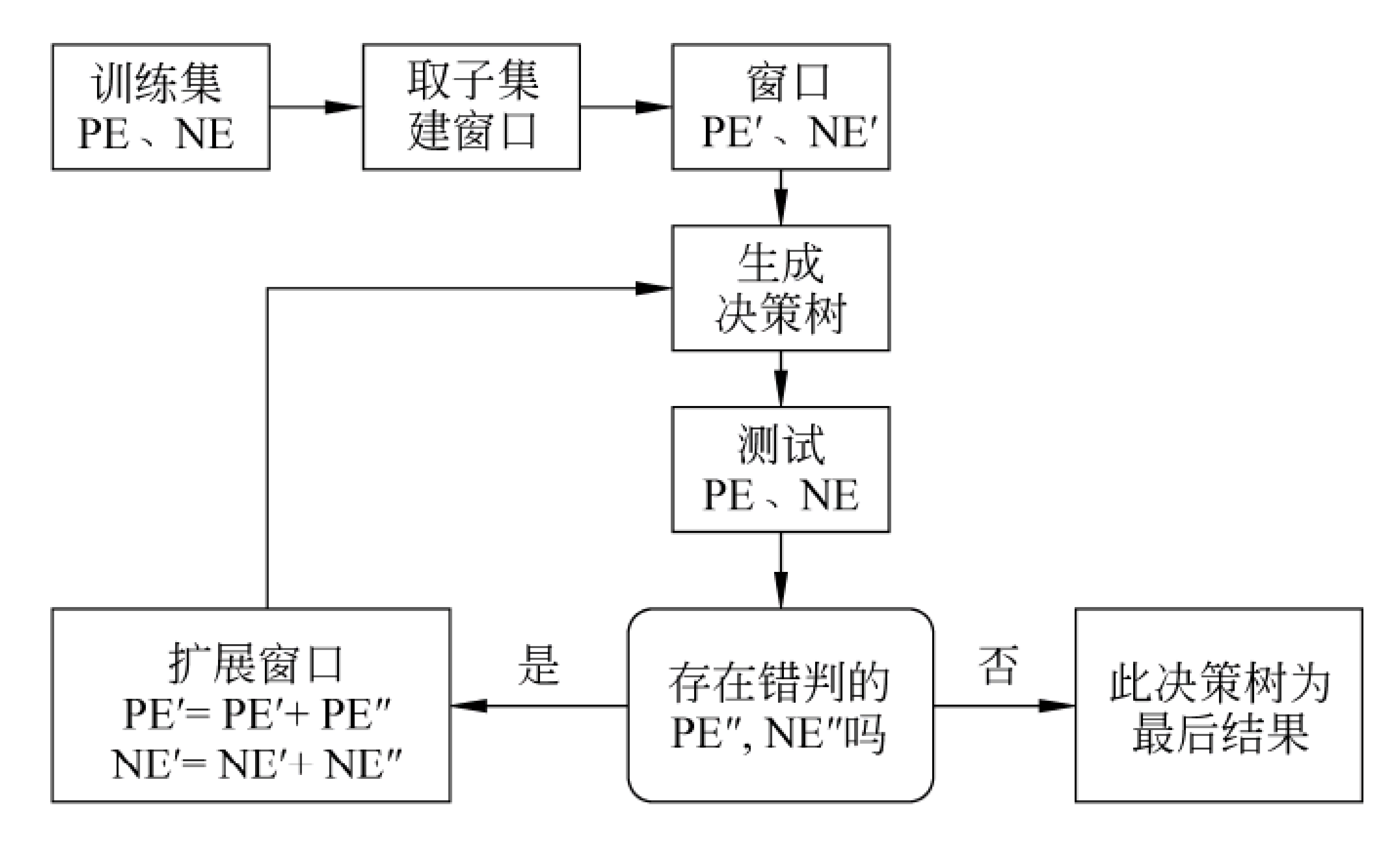



图 7.5　ID3 主算法流程

**2. 建树算法**

(1) 对当前例子集合，计算各特征的互信息。

(2) 选择互信息最大的特征 $A_k$，作为树(或子树)的根结点。

(3) 把在 $A_k$ 处取值相同的例子归于同一子集，该取值作为树的分支。$A_k$ 取几个值就得几个子集，各取值作为树的一个分支。

(4) 对既含正例又含反例的子集，递归调用建树算法。

(5) 若子集仅含正例或反例，对应分支标上 $P$ 或 $N$，返回调用处。

## 7.2.4　实例与讨论

**1. 实例计算**

对于气候分类问题进行具体计算有：

(1) 信息熵的计算

信息熵：$H(U)=-\sum_i P(u_i)\log_2 P(u_i)$

类别 $u_i$ 出现概率：$P(u_i)=|u_i|/|S|$

$|S|$表示例子集 $S$ 的总数，$|u_i|$表示类别 $u_i$ 的例子数。

对 9 个正例和 5 个反例有：$P(u_1)=9/14$，$P(u_2)=5/14$

$$H(U)=(9/14)\log_2(14/9)+(5/14)\log_2(14/5)=0.94\text{bit}$$

(2) 条件熵计算

条件熵：$H(U/V)=-\sum_j P(v_j)\sum_i P(u_i/v_j)\log_2 P(u_i/v_j)$

属性 $A_1$ 取值 $v_j$ 时，类别 $u_i$ 的条件概率：$P(u_i/v_j)=|u_i|/|v_j|$

$A_1$＝天气，它的取值有：$v_1$＝晴，$v_2$＝多云，$v_3$＝雨

在 $A_1$ 处取值“晴”的例子 5 个，取值“多云”的例子 4 个，取值“雨”的例子 5 个，故：

$$P(v_1)=5/14\quad P(v_2)=4/14\quad P(v_3)=5/14$$

取值为“晴”的 5 个例子中有 2 个正例、3 个反例，故：

$$P(u_1/v_1) = 2/5,\quad P(u_2/v_1) = 3/5$$

取值为“多云”时有：$P(u_1/v_2)=4/4$，$P(u_2/v_2)=0$

取值为“雨”时有：$P(u_1/v_3)=2/5$，$P(u_2/v_3)=3/5$

$$H(U/V) = (5/14)((2/5)\log(5/2) + (3/5)\log(5/3)) + (4/14)((4/4)\log(4/4) + 0) + (5/14)((2/5)\log(5/2) + (3/5)\log(5/3)) = 0.694\text{bit}$$

(3) 互信息计算

对 $A_1$＝天气：

$$I(\text{天气}) = H(U) - H(U \mid V) = 0.94 - 0.694 = 0.246\text{bit}$$

类似可得：

$$I(\text{气温}) = 0.029\text{bit}$$
$$I(\text{湿度}) = 0.151\text{bit}$$
$$I(\text{风}) = 0.048\text{bit}$$

(4) 建决策树的树根和分支

ID3 算法将选择互信息最大的特征“天气”作为树根，在 14 个例子中对“天气”的 3 个取值进行分支，3 个分支对应 3 个例子的子集，例子的编号分别是：

$$F1=\{1,2,8,9,11\},\quad F2=\{3,7,12,13\},\quad F3=\{4,5,6,10,14\}$$

其中 $F2$ 中的例子全属于 $P$ 类，因此对应分支标记为 $P$，其余两个子集既含有正例又含有反例，将递归调用建树算法。

(5) 递归建树

分别对 $F1$ 和 $F3$ 子集利用 ID3 算法，在每个子集中对各特征（仍为四个特征）求互信息。

① $F1$ 中的“天气”全取“晴”值，则 $H(U)=H(U|V)$，有 $I(U|V)=0$，在余下三个特征中求出“湿度”互信息最大，以它为该分支的根结点，再向下分支。“湿度”取“高”的例子全为 $N$ 类，该分支标记 $N$。取值“正常”的例子全为 $P$ 类，该分支标记 $P$。

② 在 $F3$ 中，对四个特征求互信息，得到“风”特征互信息最大，则以它为该分支根结点。再向下分支，“风”取“有风”时，例子子集全为 $N$ 类，该分支标记 $N$。取“无风”时，例子子集全为 $P$ 类，该分支标记 $P$。

这样就得到图 7.4 所示的决策树。

### 2. 对 ID3 的讨论

(1) 优点

ID3 在选择重要特征时利用了互信息的概念，算法的基础理论清晰，使得算法较简单，是一个很有实用价值的示例学习算法。

该算法的计算时间是例子个数、特征个数、结点个数之积的线性函数。钟鸣曾用 4761 个关于苯的质谱例子做了试验。其中正例 2361 个，反例 2400 个，每个例子由 500 个特征描述，每个特征取值数目为 6，得到一棵有 1514 个结点的决策树。对正、反例各 100 个测试例作了测试，正例判对 82 个，反例判对 80 个，总预测正确率 81%，效果是令人满意的。

（2）缺点

① 互信息的计算依赖于特征取值的数目较多的特征，这样不太合理。一种简单的办法是对特征进行分解，如上节例中，特征取值数目不一样，可以把它们统统化为二值特征，如天气取值晴、多云、雨，可以分解为三个特征；天气—晴，天气—多云，天气—雨。取值都为“是”或“否”，对气温也可做类似的工作。这样就不存在偏向问题了。

② 用互信息作为特征选择量存在一个假设，即训练例子集中（只有 14 个例子）的正，反例的比例应与实际问题领域里（例子数会很大）正、反例比例相同。一般情况不能保证相同，这样计算训练集的互信息就有偏差。

③ ID3 在建树时，每个结点仅含一个特征，是一种单变量的算法，特征间的相关性强调不够。虽然它将多个特征用一棵树连在一起，但联系还是松散的。

④ ID3 对噪声较为敏感。关于什么是噪声，Quinlan 的定义是训练例子中的错误就是噪声。它包含两方面，一是特征值取错，二是类别给错。

⑤ 当训练集增加时，ID3 的决策树会随之变化。在建树过程中，各特征的互信息会随例子的增加而改变，从而使决策树也变化。这对渐近学习（即训练例子不断增加）是不方便的。

总的来说，ID3 由于其理论的清晰、方法简单、学习能力较强，适于处理大规模的学习问题，在世界上广为流传，得到了极大的关注，是数据挖掘和机器学习领域中的一个极好范例，也不失为一种知识获取的有用工具。

### 7.2.5 C4.5 方法

ID3 算法在数据挖掘中占有非常重要的地位。但是，在应用中，ID3 算法存在不能够处理连续属性、计算信息增益时偏向于选择取值较多的属性等不足。C4.5 是在 ID3 的基础上发展起来的决策树生成算法，由 J. R. Quinlan 于 1993 年提出。C4.5 克服了 ID3 在应用中存在的不足，主要体现在以下几个方面：

（1）用信息增益率来选择属性，它克服了用信息增益选择属性时偏向选择取值多的属性的不足；

（2）在树构造过程中或者构造完成之后，进行剪枝；

（3）能够完成对连续属性的离散化处理；

（4）能够对于不完整数据进行处理，例如未知的属性值；

（5）C4.5 采用的知识表示形式为决策树，并最终可以形成产生式规则。

#### 1. 构造决策树

设 $T$ 为数据集，类别集合为$\{C_1, C_2, \cdots, C_k\}$，选择一个属性 $V$ 把 $T$ 分为多个子集。设 $V$ 有互不重合的 $n$ 个取值$\{v_1, v_2, \cdots, v_n\}$，则 $T$ 被分为 $n$ 个子集 $T_1, T_2, \cdots, T_n$，这里 $T_i$ 中的所有实例的取值均为 $v_i$。

令$|T|$为数据集 $T$ 的例子数，$|T_i|$为 $v=v_i$ 的例子数，$|C_j| = \text{freq}(C_j, T)$，为 $C_j$ 类的例子数，$|C_j^v|$是 $V=v_i$ 例子中，具有 $C_j$ 类别例子数。

则有：

（1）类别 $C_j$ 的发生概率：$p(C_j)=|C_j|/|T|=\text{freq}(C_j,T)/|T|$

（2）属性 $V=v_i$ 的发生概率：$p(v_i)=|T_i|/|T|$

（3）属性 $V=v_i$ 的例子中，具有类别 $C_j$ 的条件概率：$p(C_j|v_i)=|C_j^v|/|T_i|$

Quinlan 在 ID3 中使用信息论中的信息增益(gain)来选择属性，而 C4.5 采用属性的信息增益率(gain ratio)来选择属性。

以下公式中的 $H(C)$、$H(C/V)$、$I(C,V)$、$H(V)$是信息论中的写法，而 $\text{info}(T)$、$\text{info}_v(T)$、$\text{gain}(V)$、$\text{plit_info}(V)$、gain_ratio 是 Quinlan 的写法。在此统一起来。

（1）类别的信息熵

$$H(C)=-\sum_j p(C_j)\log(p(C_j))=-\sum_j \frac{|C_j|}{|T|}\log\left(\frac{|C_j|}{|T|}\right)$$
$$=-\sum_{j=1}^{k}\frac{\text{freq}(C_j,T)}{|T|}\times\log_2\left(\frac{\text{freq}(C_j,T)}{|T|}\right)=\text{info}(T)$$

（2）类别条件熵

按照属性 $V$ 把集合 $T$ 分割，分割后的类别条件熵为

$$H(C\mid V)=-\sum_j p(v_j)\sum_i p(C_j\mid v_i)\log p(C_j\mid v_i)=-\sum_j\frac{|T_j|}{|T|}\sum_i\frac{|C_j^v|}{|T_i|}\log\frac{|C_j^v|}{|T_i|}$$
$$=\sum_{i=1}^{n}\frac{|T_i|}{|T|}\times\text{info}(T_i)=\text{info}_v(T)$$

（3）信息增益(gain)，即互信息

$$I(C,V)=H(C)-H(C\mid V)=\text{info}(T)-\text{info}_v(T)=\text{gain}(V)$$

（4）属性 $V$ 的信息熵

$$H(V)=-\sum_i p(v_i)\log(p(v_i))=-\sum_{i=1}^{n}\frac{|T_i|}{|T|}\times\log_2\left(\frac{|T_i|}{|T|}\right)=\text{split_info}(V)$$

（5）信息增益率

$$\text{gain_ratio}=I(C,V)/H(V)=\text{gain}(V)/\text{split_info}(V)$$

C4.5 对 ID3 改进是用信息增益率来选择属性。

理论和实验表明，采用“信息增益率”(C4.5 方法)比采用“信息增益”(ID3 方法)更好，主要是克服了 ID3 方法选择偏向取值多的属性。

**2. 连续属性的处理**

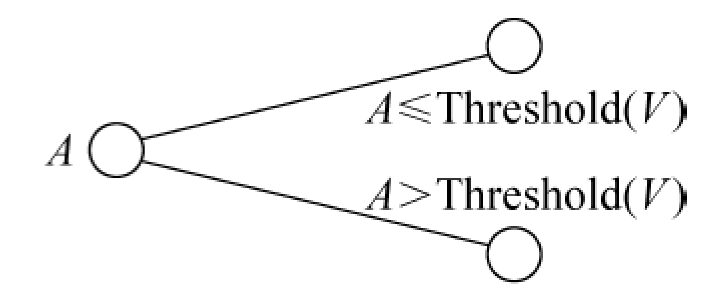
在 ID3 中没有处理连续属性的功能。在 C4.5 中，设在集合 $T$ 中，连续属性 $A$ 的取值为$\{v_1,v_2,\cdots,v_m\}$，则任何在 $v_i$ 和 $v_{i+1}$之间的任意取值都可以把实例集合分为两部分 $T_1=\{t|A\leqslant v_i\}$和 $T_2=\{t|A>v_i\}$。

可以看到一共有 $m-1$ 种分割情况，对属性 $A$ 的 $m-1$ 种分割的任意一种情况，作为该属性的两个离散取值，重新构造该属性的离散值，再按照上述公式计算每种分割所对应的信息增益率 $\text{gain_ratio}(v_i)$，在 $m-1$ 种分割中，选择最大增益率的分割作为属性 $A$ 的分支，即

$$\text{Threshold}(V)=v_k$$

其中，$\text{gain_ratio}(v_k)=\max\{\text{gain_ratio}(v_i)\}$，即 $v_k$ 是各 $v_i$ 的信息增益率最大者。

则连续属性 $A$ 可以分割为




**3. 决策树剪枝**

由于噪声和随机因素的影响,决策树一般会很复杂,因此需要进行剪枝操作。

(1) 什么时候剪枝

有两种剪枝策略:①在树生成的过程中判断是否还继续扩展决策树。若停止扩展,则相当于剪去该结点以下的分支。②对于生成好的树剪去某些结点和分支。C4.5 采用第二种方法。

剪枝之后的决策树的叶结点不再只包含一类实例。结点有一个类分布描述,即该叶结点属于某类的概率。

(2) 基于误差的剪枝

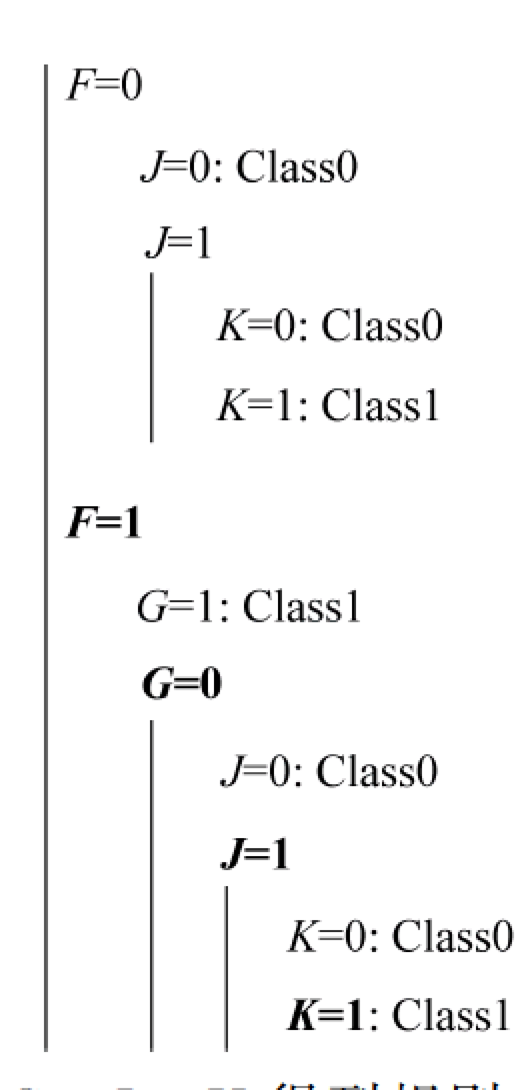
决策树的剪枝通常是用叶结点替代一个或者多个子树,然后选择出现概率最高的类作为该结点的类别。在 C4.5 中,还允许用其中的树枝来替代子树。

如果使用叶结点或者树枝代替原来的子树之后,误差率若能够下降,则使用此叶结点或者树枝代替原来的子树。

**4. 从决策树抽取规则**

在 C4.5 中,对于生成好的决策树,可以直接从中获得规则。从根到叶的每一条路径都可以是一条规则。这样,可以看出有多少条路径就可以产生多少条规则。例如,从下面的决策树中可以得到规则:

决策树:




沿着决策树其中一条路径 $F \to G \to J \to K$ 得到规则:

```
IF F=1, G=0, J=1, K=1 THEN class1
```

# 7.3 决策规则树方法

## 7.3.1 IBLE 方法基本思想

### 1. IBLE 方法的特点

钟鸣与笔者于 1991 年研制的 IBLE(Information-Based Learning from Examples)方法是基于信息论的示例学习方法，利用信息论中信道容量的概念作为对实体中选择重要特征的度量。信道容量是一个不依赖于正、反例的比例，仅依赖于训练集中正、反例的特征取值的选择量。这样，信道容量克服了互信息依赖正反例比例的缺点。IBLE 方法不同于 ID3 方法每次只选一个特征作为决策树的结点，而是选一组重要特征建立规则，作为决策树的结点。这样，用多个特征组合成规则的结点来鉴别实例，能够更有效地正确判别。对那些不能直接判定的例子继续利用决策规则树的其他规则结点来判别，这样一直进行下去，直至判出类别为止。

IBLE 方法建立的是决策规则树，树中每个结点是由多个特征所组成的。特征的选取是通过计算各特征信道容量来进行的。各特征的正例标准值由译码函数决定。结点中判别正反例的阈值($S_n$，$S_p$)是由实例中权值变化的规律来确定的。

### 2. 多元信道转化成二元信道

在各特征取多值的情况下，用互信息作为特征选择量，会出现倾向于取某值的例子数较多的特征，这种倾向并不都合理。用信道容量作为特征选择量也必然有同样的问题存在。一种解决办法是对特征进行分解，如前面举的例中，特征取值数目不一样可以把它们统统化为二值特征。例如天气取值晴，多云，雨，可以分解成三个特征：天气－晴、天气－多云、天气－雨，每个都取值为{yes，no}，对气温也可以做类似的工作。这样在选择特征时就不会出现偏向问题了。

### 3. 决策规则树

IBLE 算法从训练集中归纳出一棵决策规则树。

判定一个实体属于 $u_1$ 类还是属于 $u_2$ 类，首先从分析该实体的特征入手，用规则分析会得出三种可能结论，①该实体属于 $u_1$ 类，②该实体属于 $u_2$ 类，③不能做出判定，需进一步分析后再做结论。在进一步分析时又会出现上述三种情形。对一实体的分析，这个过程一直进行到得出具体类别为止。IBLE 就是依据这种思想构造决策规则树的。决策规则树如图 7.6 所示。

对于更复杂的问题除使用主规则外，还增加分规则，得出如图 7.7 所示的决策规则树。

### 4. 决策规则树结点

(1) 规则表示形式

决策规则树中非叶结点均为规则。规则表示为

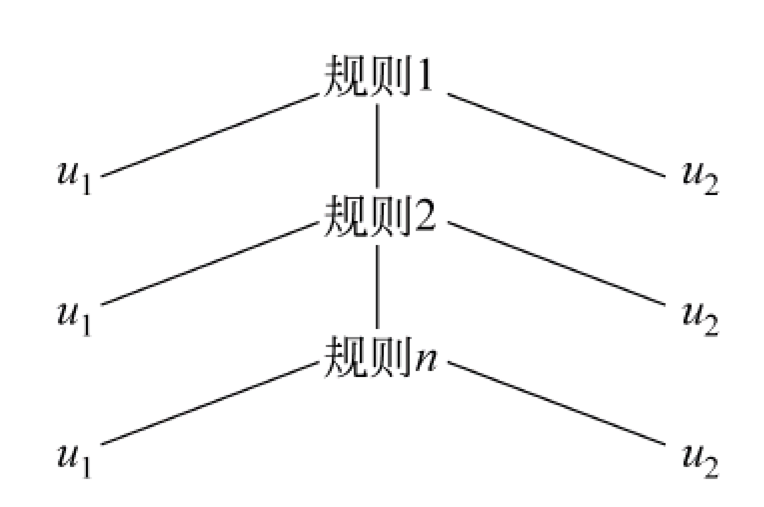


图 7.6 IBLE 算法的一般决策规则树

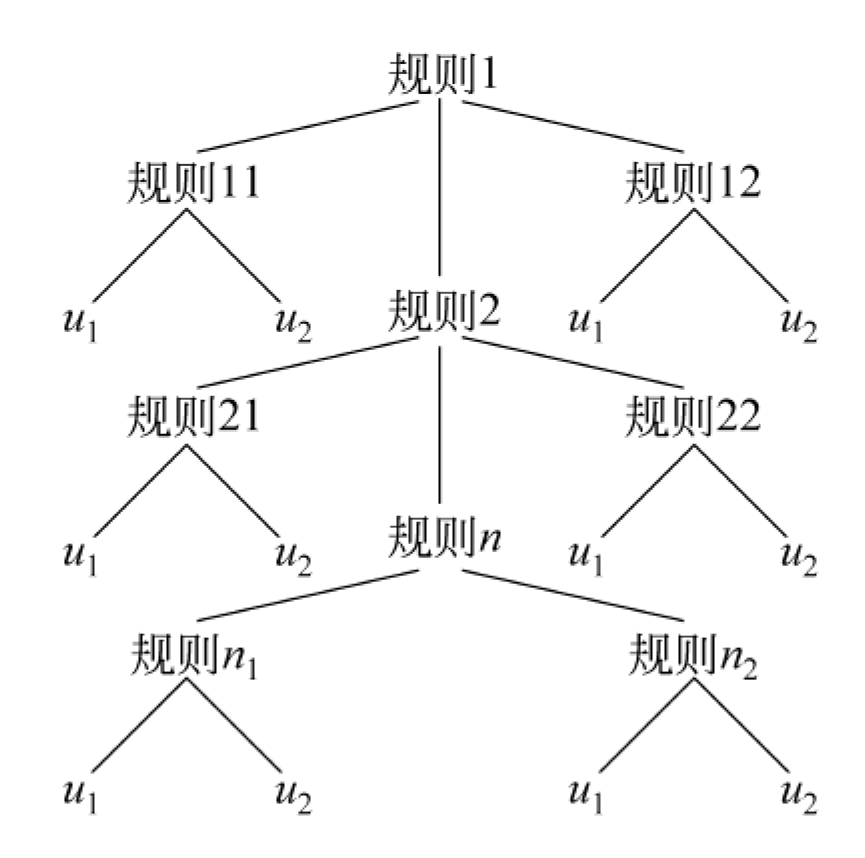


图 7.7 IBLE 算法的复杂决策规则树

特征：$A_1, A_2, \cdots, A_m$

权值：$W_1, W_2, \cdots, W_m$

标准值：$V_1, V_2, \cdots, V_m$

阈值：$S_p, S_n$

该规则可形式描述为

① sum:=0；

② 对 $i:=1$ 到 $m$ 作：若$(A_i)=V_i$，则 sum:=sum+$w_i$；

③ 若 sum≤$S_n$，则该例为 $N$ 类；

④ 若 sum≥$S_p$，则该例为 $P$ 类；

⑤ 若 $S_n$<sum<$S_p$，则该例暂不能判，转下一条规则判别。

其中 sum 表示权和，$(A_i)$表示特征 $A_i$ 的取值。

规则说明：$A_1, A_2, \cdots, A_m$ 为组成规则的特征，$W_1, W_2, \cdots, W_m$ 为对应的权值，$V_1, V_2, \cdots, V_m$ 为对应特征取正例的标准值，若例子在该特征处取值与标准值相同，则 sum(权和)加上对应权值，否则不加。$S_p$、$S_n$ 是判是、判非、不能判的阈值。若例子的权和为 sum，sum≥$S_p$ 时判为是类($u_1$ 类)，sum≤$S_n$ 时判为非类($u_2$ 类)，$S_n$<sum<$S_p$ 时认为不能判。由于 $S_p$、$S_n$ 的作用知道，图 7.7 的分规则中必有 $S_p=S_n$。

(2) 举例

为说明规则中各成分的意义，举一个例子。设问题空间中例子有 10 个特征(属性)，特征编号从 1 到 10。每个特性取值为{no, yes}，用{0,1}表示，规则是由重要特征组成的，对每个特征求出权值以表示其重要程度，删除不重要特征得规则如下：

| | | | | | |
|---|---|---|---|---|---|
| 特征： | 1 | 3 | 4 | 6 | 7 |
| 权值： | 100 | 90 | 105 | 50 | 40 |
| 标准值： | 1 | 0 | 1 | 1 | 0 |

阈值： 220,100

现有三个测试例子：

例子 1：(1,0,0,0,1,0,0,1,1,1)

例子 2：(0,1,0,0,1,0,0,0,1,0)

例子 3：(0,1,0,0,1,0,1,0,1,1)

例子 1 的权和 sum=230,有 sum>220,判定例子 1 属于 $u_1$ 类。例子 2 的权和 sum=130,有 100<sum<220,认为例子 2 不能判,而例子 3 有权和 sum=90,有 sum<100,判例子 3 的类别为 $u_2$ 类。

IBLE 算法由四部分组成：预处理、建决策树算法、建规则算法、类别判定算法。下面分别介绍。

## 7.3.2 IBLE 算法

### 1. 预处理

将例子集的特征取多值,变为多个特征分别取{0,1}值,即一个特征取 $n$ 个值变为 $n$ 个特征分别取{0,1}值。

### 2. 建规则算法

- 求各特征 $A_k$ 的信道容量 $C_k$,对于一个特征有分特征(原一个特征取多值变成多个特征取{0,1}值时,该多个特征为原特征的分特征)时,取最大 $C$ 值的分特征代表该特征。

权值的计算(取整)公式为：$W_k=[C_k\times1000]$。

- 利用最大后验概率准则定义该特征 $A_k$ 的译码函数 $F(1)$、$F(0)$。

设类别为 $u_1,u_2$,特征 $V$ 取值 1 和 0,转移概率为 $P(1/u_1),P(0/u_1),P(1/u_2),P(0/u_2)$。信道容量计算后,可同时得到类别的先验概率 $P(u_1)$和 $P(u_2)$。于是,令

$$\mathrm{SUM}=P(u_1)\times P(1/u_1)+P(u_2)\times P(1/u_2)$$

由贝叶斯公式：$P(u_1/1)=P(u_1)\times P(1/u_1)/\mathrm{SUM}$,

$$P(u_2/1)=P(u_2)\times P(1/u_2)/\mathrm{SUM}$$

译码准则为：当 $P(u_1/1)\geqslant P(u_2/1)$时,$F(1)=u_1$;否则,$F(0)=u_1$。这样,就定义了特征 $V$ 对类别 $u_1$(正例)的标准值 1 或 0。可以证明,该准则的错误概率最小。

- 利用译码函数按正例($u_1$)输入,计算特征 $A_k$ 的标准值{0,1}。
- 选取前 $m$ 个信道容量(即权值)较大的特征构造规则。

一般说来,$m$ 的选取应保证 $C>0.01$bit 的特征都被选中(对具体问题可通过试验来确定)。

- 计算所有的正反例的权和数,从它们的分布规律中得出 $S_p,S_n$ 阈值。

建立一个二维数组 $A(m,n)$,$m=1,2,3;n=1,2,\cdots,|U|$($|U|$表示例子总数)。它由三项组成,$A(1,n)$存放各例的权和(例子中各特征的权值累加之和),$A(2,n)$存放正例个数,当例子是正例时,它为 1,反之为零。$A(3,n)$存放反例个数,当例子是反例时,它为 1,反之为零。

先对各正,反例子求权和并填入数组 $A(m,n)$中。再按权和大小从小到大的顺序对数组 $A(m,n)$进行排序,对权和相同的不同的正反例,将它们合并成一列相同的权和,累计正反例个数。这样,数组缩小了,即 $n\leqslant|U|$。而且正反例权和的规律性就出现了：权和小的

部分，正例个数为零，反例个数偏大；权和大的部分，正例个数偏大，反例个数为零，如图 7.8 所示。

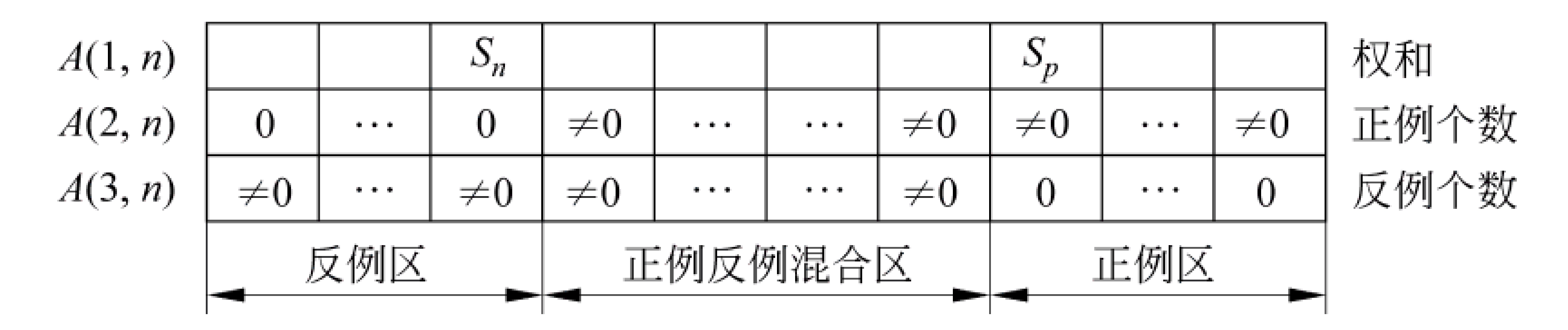



图 7.8　正、反例权和变化规律

从图 7.8 中可知，整个例子集合中，划分成三个区：反例区，正反例混合区，正例区。在反例区中，正例个数 $A(2,n)$均为零。在正例区中，反例个数 $A(3,n)$均为零。在混合区中，正例个数 $A(2,n)$和反例个数 $A(3,n)$均不为零。在三个区的分界线处的权和值作为 $S_p$、$S_n$ 值，用做判别正反例的阈值。

**3. 建决策树算法**

设 $T$ 为存放决策规则树的空间。

(1) 置决策规则树 $T$ 为空。分配一新结点 $R$，$T:=R$。

(2) 对当前训练集 PE∪NE，利用“建规则算法”构造主规则。

(3) 用当前规则测试 PE、NE 得子集 PEP、PEN、PEM(正例三个子集)，NEP、NEN、NEM(反例三个子集)。其中 PEP、PEN、PEM 分别表示正例被判为：$P$ 类、$N$ 类、不能判这三个子集。NEP、NEN、NEM 分别表示反例被判为：$P$ 类、$N$ 类、不能判这三个子集。

(4) 将当前规则放入结点 $R$。

(5) 若$(|\text{PEP}|\neq 0)\vee(|\text{NEP}|\neq 0)$则 PE：=PEP；NE：=NEP；分配一新结点 $W_1$；$R$ 左指针指向 $W_1$。

① 对当前训练集 PE∪NE 利用“建规则算法”构造左分规则；

② 将左分规则放入结点 $W_1$。

(6) 若$(|\text{PEN}|\neq 0)\ \vee|\text{NEN}|\neq 0)$则 PE：=PEN，NE：=NEN；分配一新结点 $W_2$；$R$ 右指针指向 $W_2$。

① 对当前训练集 PE∪NE 利用“建规则算法”构造右分规则；

② 将右分规则放入结点 $W_2$。

(7) 若$(|\text{PEM}|\neq 0)\vee(|\text{NEM}|\neq 0)$则 PE：=PEM，NE：=NEM；分配一新结点 $W_3$；$R$ 的中指针指向 $W_3$；$R:=W_3$；转(2)。

(8) 结束。

建决策树算法如图 7.9 所示。

**4. 类别判定算法**

在得到一棵决策规则树后，对一未知实体 $E$ 如何分类，下面给出具体的算法：

(1) 置根结点为当前结点。

(2) 用当前结点中的规则对 $E$ 进行判定。

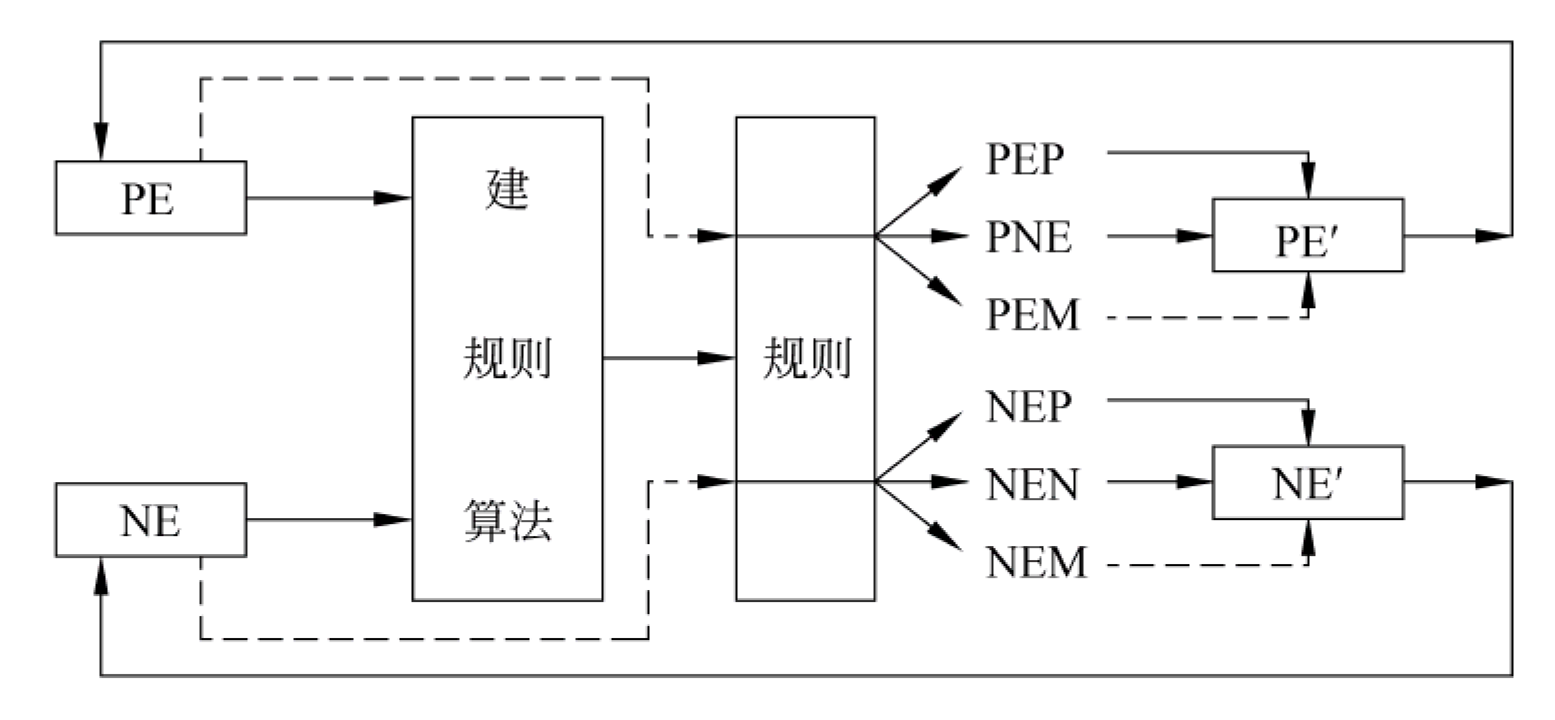


图 7.9　IBLE 建决策树算法图

① 判为 $P$ 时(对主规则,该实体不一定是 $P$ 类),若当前结点左指针不空(即左规则存在),将左指针指示的结点置为当前结点且转(2),否则(左指针为空,该实体判为 $P$ 类)转(3)。

② 判为 $N$ 时(对主规则,该实体不一定是 $N$ 类),若当前结点右指针不为空(即右规则存在),则将右指针指示的结点置为当前结点且转(2),否则(右指针为空,该实体判为 $N$ 类)转(3)。

③ 不能判时,将当前结点的中指针指示的结点置为当前结点转(2)。

(3) 输出判别结果,结束。

## 7.3.3　IBLE 方法实例

### 1. 配隐形眼镜问题

(1) 简例说明

① 患者配隐形眼镜的类别

患者是否应配隐形眼镜有三类:

@1:患者应配隐形眼镜;

@2:患者应配软隐形眼镜;

@3:患者不适合配隐形眼镜。

② 患者眼镜诊断信息(属性)

a:患者的年纪

年轻;前老光眼;老光眼

b:患者的眼睛诊断结果

近视;远视

c:是否散光

是:否

d:患者的泪腺

不发达;正常

③ 配隐形眼镜实例

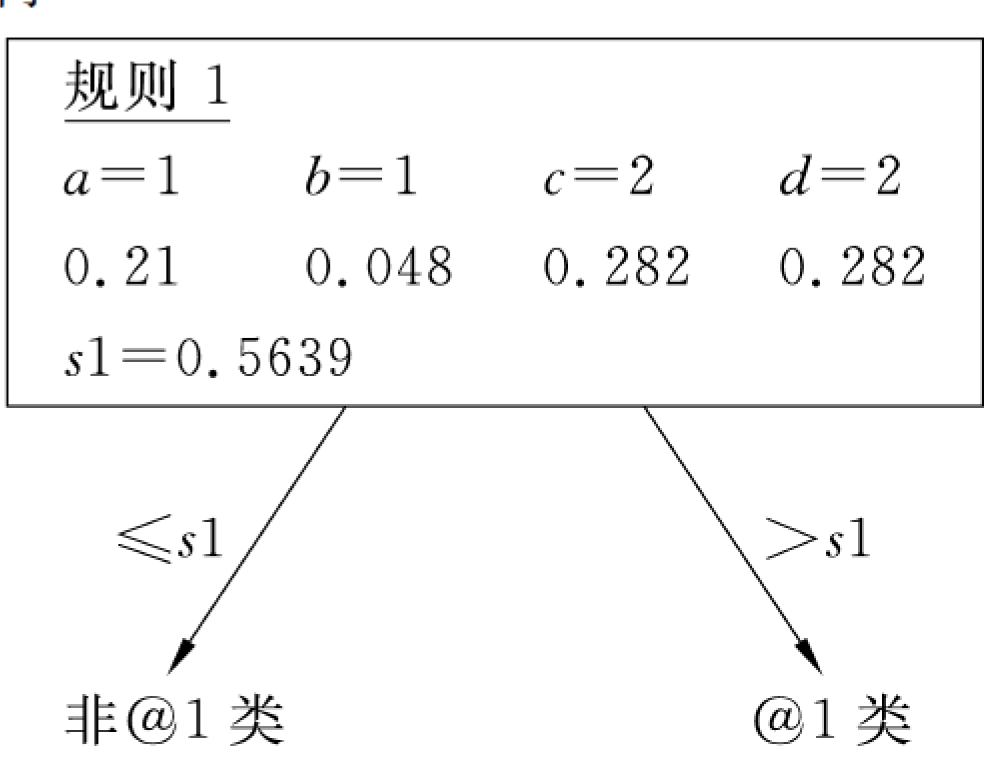
现有 24 个患者实例分别属于三个类别,如表 7.2 所示。

表 7.2 配隐形眼镜患者实例

| 序　号 | 属性取值 | 诊断值 | 序　号 | 属性取值 | 诊断值 |
|---|---|---|---|---|---|
| | $a$　$b$　$c$　$d$ | @ | | $a$　$b$　$c$　$d$ | @ |
| 1 | 1　1　1　1 | 3 | 13 | 2　2　1　1 | 3 |
| 2 | 1　1　1　2 | 2 | 14 | 2　2　1　2 | 2 |
| 3 | 1　1　2　1 | 3 | 15 | 2　2　2　1 | 3 |
| 4 | 1　1　2　2 | 1 | 16 | 2　2　2　2 | 3 |
| 5 | 1　2　1　1 | 3 | 17 | 3　1　1　1 | 3 |
| 6 | 1　2　1　2 | 2 | 18 | 3　1　1　2 | 3 |
| 7 | 1　2　2　1 | 3 | 19 | 3　1　2　1 | 3 |
| 8 | 1　2　2　2 | 1 | 20 | 3　1　2　2 | 1 |
| 9 | 2　1　1　1 | 3 | 21 | 3　2　1　1 | 3 |
| 10 | 2　1　1　2 | 2 | 22 | 3　2　1　2 | 2 |
| 11 | 2　1　2　1 | 3 | 23 | 3　2　2　1 | 3 |
| 12 | 2　1　2　2 | 1 | 24 | 3　2　2　2 | 3 |

（2）利用 IBLE 算法得出的各类决策规则树和逻辑公式

① @1 类的决策规则树

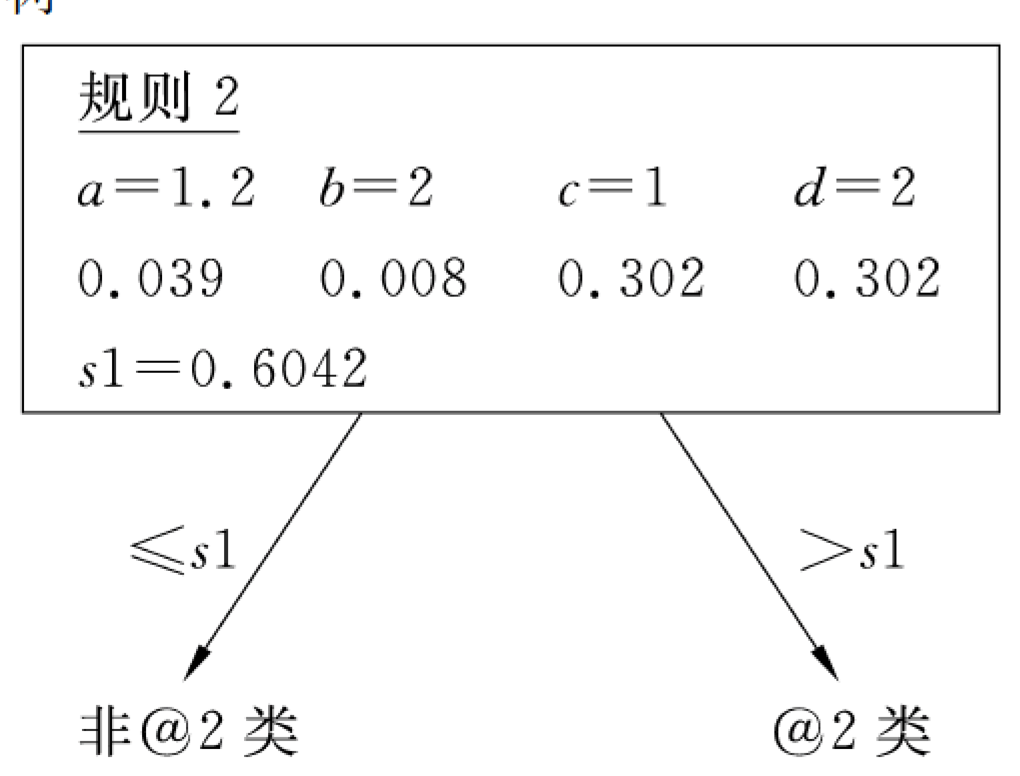



相应的逻辑公式为

$$c=2 \wedge d=2 \wedge a=1 \rightarrow @1$$

$$c=2 \wedge d=2 \wedge b=1 \rightarrow @1$$

② @2 类的决策规则树



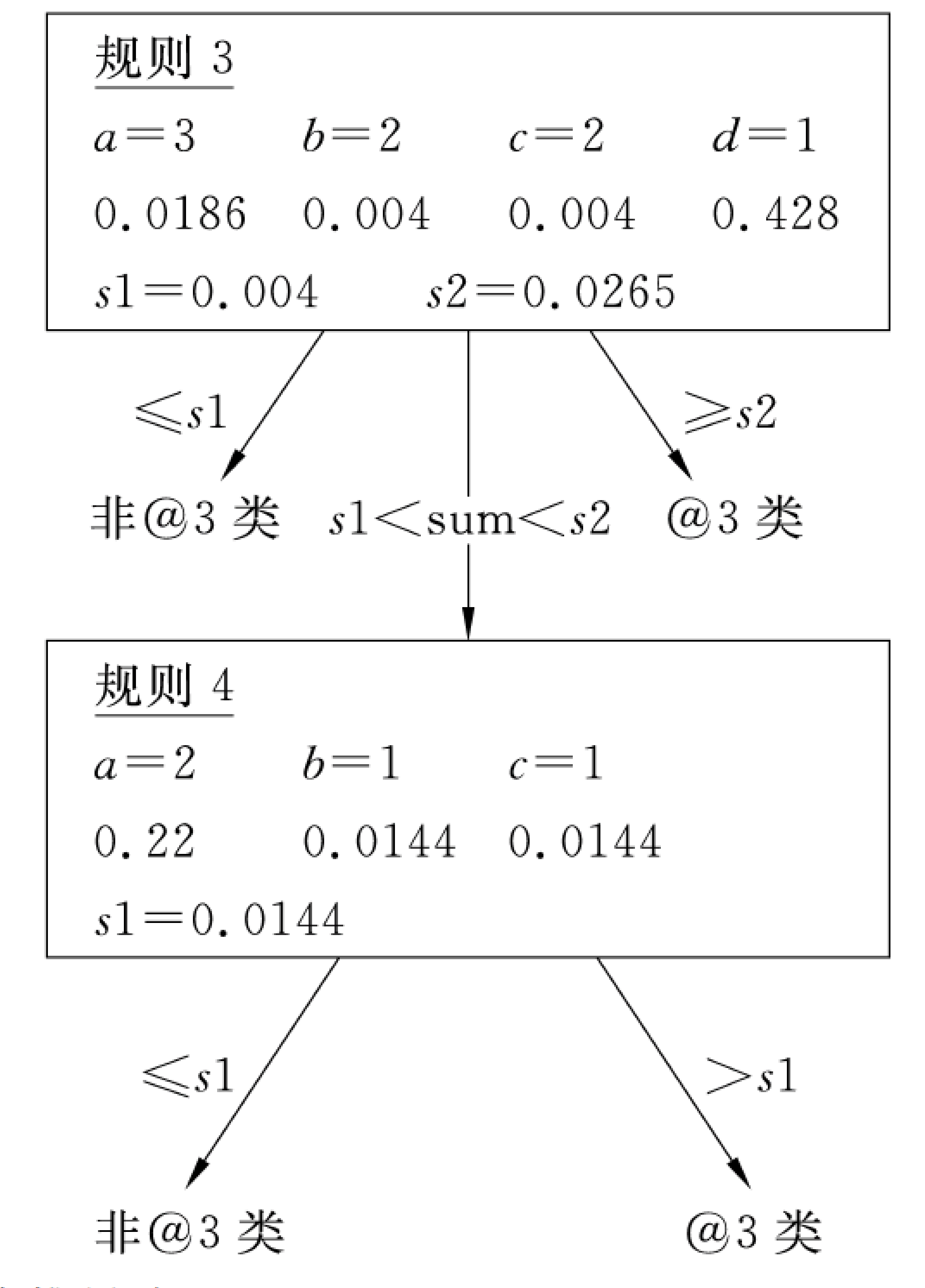
相应的逻辑公式为

$$c=1 \wedge d=2 \wedge b=2 \rightarrow @2$$

$$c=1 \wedge d=2 \wedge a=1 \rightarrow @2$$

$$c=1 \wedge d=2 \wedge a=2 \rightarrow @2$$

③ @3 类的决策规则树




该决策树的逻辑公式推导为：

- 上层结点的逻辑公式

$$d = 1 \rightarrow @3$$

$$a = 3 \wedge b = 2 \wedge c = 2 \rightarrow @3$$

- 上层不能判断逻辑公式（中线结论）

$$(b = 2 \wedge c = 2) \vee$$

$$(a = 3) \vee$$

$$(a = 3 \wedge b = 2) \vee$$

$$(a = 3 \wedge c = 2) \rightarrow \text{继续判别}$$

- 下层结点的逻辑公式

$$b = 1 \wedge c = 1 \rightarrow @3$$

$$a = 2 \rightarrow @3$$

- 合并后下层结点的逻辑公式（上层“继续判别”逻辑公式与下层结点的逻辑公式的合并。合并时，同一个变量不能同时取两个值）

$$a = 3 \wedge b = 1 \wedge c = 1 \rightarrow @3$$

$$a = 2 \wedge b = 2 \wedge c = 2 \rightarrow @3$$

### 2. 苯等八类化合物的分类问题

（1）质谱分析

质谱仪是一种化学分析仪器，它以高速电子轰击被测样本，使分子产生分裂碎片且重新排列，测量这些碎片的荷质比及能量形成质谱，如图 7.10 所示。分析化学家根据质谱可以推测出样本的分子结构及性质。这是一个极为复杂和困难的任务，原因在于质谱数据量太大且伴随噪声，而且质谱测定理论尚不完备。在这样的背景下，要用传统的知识获取技术建造一个质谱解析专家系统是极为困难的。因此，用计算机从大量的质谱数据中自动获得一些知识便成了一个诱人的设想。

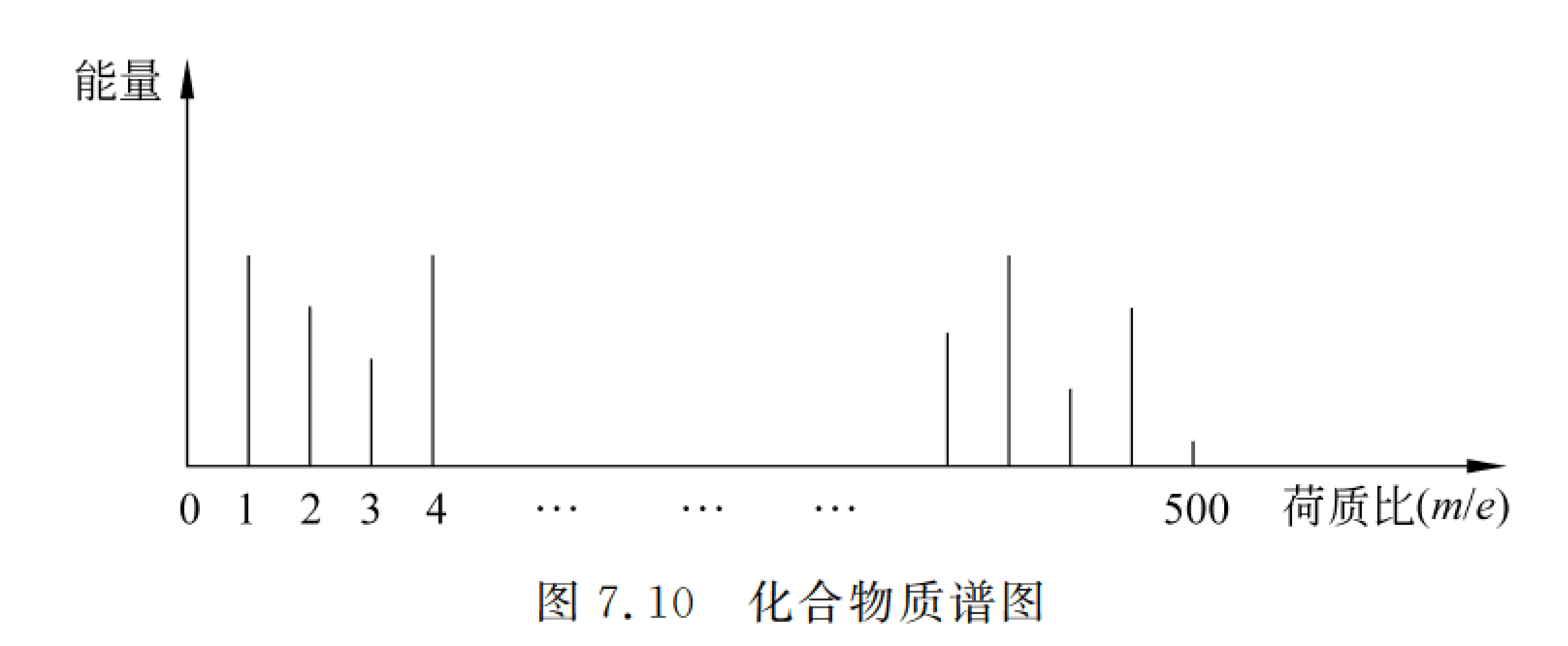



图 7.10　化合物质谱图

（2）实例计算

对八种类型的化合物进行学习、识别，其中前三种类型分别为 WLN 码中含 R、T60TJ 和 QR 的化合物；后五种为日内瓦国际会议的技术报告中给出的五类有机磷化合物，前三种类型化合物的训练集、测试集的构造方法是：从 31231 例质谱中选出某类所有化合物的集合 $T1$，剩余的两类成为集合 $T2$。从 $T1$ 中随机抽出一定数目的化合物构成两个集合 $T11$、$T12$，再从 $T2$ 中随机抽取一定数目的化合物构成两个集合 $T21$、$T22$ 用 $T11$ 和 $T21$ 组成训练集，正例 PE＝$T11$，反例 NE＝$T21$，用 $T12$ 和 $T22$ 组成测试集。对于后五种有机磷化合物（例子数不多），上述 31231 例前三类质谱中都没有，对五种类化合物输入时，每种抽取八例作为训练集中的正例集，剩下的作为测试集的正例，再从 31231 例质谱中抽出 999 例作为训练集反例集，得出如表 7.3 所示的训练集和测试集。用 IBLE 学习后得出八棵决策规则树（在此省略），对测试集进行识别，预测正确率如表 7.4 所示。

**表 7.3　八类训练物的训练集和测试集**

| 类 | 训练集 | | 测试集 | |
|---|---|---|---|---|
| | 正例 | 反例 | 正例 | 反例 |
| R | 2363 | 2400 | 102 | 155 |
| QR | 571 | 2000 | 20 | 100 |
| T60TJ | 500 | 2300 | 50 | 50 |
| 类一 | 8 | 999 | 5 | 999 |
| 类二 | 8 | 999 | 5 | 999 |

续表

| 类 | 训练集 | | 测试集 | |
|---|---|---|---|---|
| | 正例 | 反例 | 正例 | 反例 |
| 类三 | 8 | 999 | 2 | 999 |
| 类四 | 8 | 999 | 4 | 999 |
| 类五 | 8 | 999 | 1 | 999 |

**表 7.4　IBLE 对八类化合物的预测结果**

| 类 | 正例 | 认对 | 认错 | 正确百分比 | 反例 | 认对 | 认错 | 正确百分比 | 总正确百分比 |
|---|---|---|---|---|---|---|---|---|---|
| R | 102 | 95 | 7 | 93.137 | 155 | 136 | 19 | 87.774 | 90.439 |
| QR | 20 | 15 | 5 | 75 | 100 | 84 | 16 | 84 | 79.5 |
| T60TJ | 50 | 34 | 16 | 68 | 50 | 48 | 2 | 96 | 82 |
| 类一 | 5 | 5 | 0 | 100 | 999 | 997 | 2 | 99.8 | 99.9 |
| 类二 | 5 | 5 | 0 | 100 | 999 | 997 | 2 | 99.8 | 99.9 |
| 类三 | 2 | 2 | 0 | 100 | 999 | 999 | 0 | 100 | 100 |
| 类四 | 4 | 4 | 0 | 100 | 999 | 999 | 0 | 100 | 100 |
| 类五 | 1 | 1 | 0 | 100 | 999 | 999 | 0 | 100 | 100 |

本实验中，预测正确率是这样计算的：先分别计算正、反例的预测正确率，然后两者相加除以 2 得出总预测正确率，这种做法在实际问题中可信程度较高。从表 7.5 知道，对于八类化合物，IBLE 的平均预测正确率为 93.967%。

(3) IBLE 与 ID3 的比较

① 实例计算情况

为了比较 IBLE 与 ID3 在正、反例数目变化情况下的性能，从八种类型中随机抽取三类，即 R，T60TJ 和有机磷化合物中的第二类进行实验。两种算法关于三种化合物的平均预测正确率如表 7.5 所示。可以看出，IBLE 的预测正确率比 ID3 高出近 10 个百分点。

**表 7.5　IBLE 和 ID3 的平均预测正确率**

| 类 | IBLE(%) | ID3(%) |
|---|---|---|
| R | 81.779 | 72.203 |
| T60TJ | 76.786 | 70.643 |
| 类二 | 98.334 | 89.322 |

对 IBLE 算法，在训练集中正、反例子数目做大的变化时，进行测试情况见表 7.6。从表 7.6 中可见，正例数不变化，反例数逐步减少时，正确识别率稍有提高。而反例数不变，正例数减少时，正确识别率显著下降。正、反例都下降时，正确识别率在逐步下降。

**表 7.6  R 类例子数目变化时识别情况**

| 训 练 集 | | 对 正 例 | | | 对 反 例 | | |
|---|---|---|---|---|---|---|---|
| 正例 | 反例 | 认对 | 认错 | 正确% | 认对 | 认错 | 正确% |
| 2363 | 2400 | 95 | 7 | 93.137 | 84 | 18 | 82.353 |
| 2363 | 1200 | 88 | 14 | 86.275 | 84 | 18 | 82.353 |
| 2363 | 400 | 91 | 11 | 89.216 | 99 | 3 | 97.059 |
| 2363 | 200 | 98 | 4 | 96.078 | 101 | 1 | 99.1 |
| 2363 | 100 | 98 | 4 | 98.078 | 101 | 1 | 99.1 |
| 2363 | 2400 | 95 | 7 | 93.137 | 84 | 18 | 82.353 |
| 1181 | 2400 | 76 | 26 | 74.51 | 71 | 31 | 69.608 |
| 393 | 2400 | 68 | 34 | 66.667 | 46 | 56 | 45.098 |
| 196 | 2400 | 54 | 48 | 52.941 | 35 | 67 | 34.314 |
| 98 | 2400 | 50 | 52 | 49.02 | 24 | 78 | 23.520 |
| 2363 | 2400 | 95 | 7 | 93.137 | 84 | 18 | 82.353 |
| 393 | 400 | 75 | 27 | 73.529 | 75 | 27 | 73.529 |
| 196 | 200 | 87 | 15 | 85.294 | 80 | 22 | 78.431 |
| 98 | 100 | 87 | 15 | 85.294 | 70 | 32 | 68.627 |

② 原因分析

IBLE 的预测正确率之所以比 ID3 高,原因在于:

- IBLE 用信道容量作为特征选择量,而 ID3 用互信息,信道容量不依赖于正、反例的比例,互信息依赖训练集中正反例的比例。
- IBLE 在建树过程中,每次选择多个特征构成规则,变量间的相关性得到较好的体现。ID3 在建树过程中,每次选择一个特征作为结点,不能较好地体现特征间的相关性。

③ IBLE 决策规则树的特点

- IBLE 的决策规则树中的规则在表示和内容上与专家知识具有较高的一致性。

以 R(苯)的决策规则树中第一条规则为例。规则列出了峰系列,与专家知识表示是一致的,第一条规则指出在 $m/e=27,50-52,62-65,74-78,89-92,104-105$ 处应有峰。有关文献中认为含苯化合物的重要系列应是 $m/e=38-39,50-52,63-65,75-78,91,105,119,113$ 等。比较一下可知,在列出的这 16 个峰中第一条规则就包含了 12 个,而且都是权值较大的峰。专家知识中一般不指出哪些地方应无峰,而 IBLE 的规则中也指了出来,这是对专家知识的一种补充。而 ID3 的决策树在表示上与专家知识相差较大,在内容上也不易做到与专家知识具有一致性(原因在于用互信息选择主要特征依赖于训练集中正、反例的比例,而实际问题中正、反例的比例不易确定)。

- 在训练集中,若正、反例数目变化较大,IBLE 得到的规则具有较好的稳定性。

这在 R 的训练集中正、反例数目变化较大的情况下，IBLE 得出的各决策规则树中第一条规则，都含有相同的 41 个特征($m/e$＝41，42，43，50，51，54，55，56，57，58，59，62，63，64，65，67，68，69，70，71，72，75，76，77，78，81，82，83，84，85，89，90，91，92，96，97，98，100，104，105，143，包括有峰、无峰)，在相同的变化下 ID3 的决策树头两层 7 个重要能量中，无共同的特征。

总之，IBLE 的规则与专家知识在内容上有较高的一致性，用 IBLE 获取的知识建立的专家系统对实例的判别进行解释时提供了良好的条件。这一点正是 ID3 的一个重要缺陷。

显然，IBLE 比 ID3 优越。

(4) 小结

这里提出的机器学习的信道模型，系统地论述了示例学习的信息论，利用新的特征选择量一信道容量，即用信道容量来选取重要特征的思想，不仅用于机器学习和数据挖掘之中，也可以用于模式识别的特征抽取。在上面的试验中，对八类化合物的质谱分类问题，用神经网络中的感知机和反向传播模型进行学习，由于特征太多，两种方法的迭代都不收敛。

利用信道容量进行特征提取后，再用感知机和 B-P 模型学习，都取得了较好的效果。感知机的平均预测正确率为 79%，B-P 模型的平均预测正确率为 84%。

IBLE 示例学习算法实现简单、学习正确性较高，所得知识在表示和内容上与专家知识有较高的一致性，而且特别适合于处理大规模的学习问题，可作为专家系统的知识获取工具。

# 习　题　7

1. 信息论的基本原理是什么？
2. 学习信道模型是什么？
3. 为什么机器学习和数据挖掘的分类问题可以利用信息论原理？
4. 自信息和互信息的含义是什么？它们的计算公式是什么？
5. 信道容量的含义是什么？它与互信息有什么关系？
6. 译码准则的基本思想是什么？
7. 决策树方法的基本思想是什么？
8. 说明 ID3 方法的建树算法步骤。
9. 设计用 ID3 决策树进行实例判别的判定算法。
10. 编制 ID3 算法的计算机程序，并用表 7.1 气候训练集例子进行测试。
11. 对于 7.1 气候训练集，用 CLS 方法建树：任意选一字段项(如气温)为根结点，其字段项各取值为分支，对各分支数据子集重复上述操作，向下扩展此决策树，直到数据子集属于同一类数据(即叶结点)为止，并标记叶结点为 $P$ 类或 $N$ 类。

    请比较 CLS 决策树与 ID3 决策树的优缺点。
12. 在表 7.1 气候训练集中，对天气＝晴的数据子集，计算各特征(天气、气温、湿度、风)的互信息是多少？哪个特征的互信息最大？
13. C4.5 方法对 ID3 方法的改进主要体现在什么地方？

14. 信息增益率与信息增益有什么不同？在 C4.5 中为什么使用信息增益率作为分支标准？

15. 在 C4.5 中如何对连续属性进行处理？

16. IBLE 算法用什么来选择重要属性构造决策规则树结点？

17. IBLE 决策树的表示形式是什么？比较 IBLE 决策规则树和 ID3 决策树有什么不同？

18. IBLE 决策树中结点的表示形式是什么？

19. 设某例子集的 IBLE 决策规则树的结点规则为：

| 特征 | $a$ | $b$ | $c$ | $d$ |
| --- | --- | --- | --- | --- |
| 权值 | 0.021 | 0.048 | 0.282 | 0.282 |
| 标准值 | 1 | 1 | 2 | 2 |
| 阈值 | $S_n=0.564$ | | $S_p=0.585$ | |

现有两个例子的特征取值分别为：

$$a=1,\quad b=2,\quad c=2,\quad d=2$$

$$a=1,\quad b=1,\quad c=1,\quad d=2$$

请用该结点规则判别它们属于{$P$ 类、$N$ 类、不能判别}中的哪种情况？

20. 说明 IBLE 决策规则树中结点中阈值 $S_n$ 和 $S_p$ 求解的思想。

21. 说明 IBLE 建规则算法。

22. 说明隐形眼镜简例中@3 类决策规则树的含义。

23. 说明从简例中@3 类决策规则树求出其相应的逻辑公式。

24. 请说明 IBLE 方法比 ID3 方法的技术进步点。

# 第8章 集合论方法

集合论原理是数据挖掘的重要理论基础，可用于分类问题、聚类问题和关联规则挖掘。

集合论原理用于分类问题时，主要是利用集合之间的覆盖关系，如粗糙集方法是对条件属性和决策（类别）属性中的等价类（一个或多个属性取值均相同的元组）之间的覆盖关系；AQ11 方法是对覆盖正例排斥反例的种子（多个属性取值的与关系），构成规则知识。

集合论原理用于解决聚类问题时，主要是按数据集中元素间的距离远近或相似度大小，聚成多个类别集合。

集合论原理用于关联规则挖掘时是计算数据项（如商品）集在整个集合中和相关集合中所占的比例，大于阈值（支持度和可信度）时构成数据项之间关联规则。

## 8.1 粗糙集方法

### 8.1.1 粗糙集概念

粗糙集（Rough Set）是波兰数学家 Z. Pawlak 于 1982 年提出的。粗糙集以等价关系（不可分辨关系）为基础，用于分类问题。它用上、下近似两个集合来逼近任意一个集合，该集合的边界线区域被定义为上近似集和下近似集之差集。上、下近似集可以通过等价关系给出确定的描述，边界域的含糊元素数目可以被计算出来。而模糊集（Fuzzy）是用隶属度来描述集合边界的不确定性，隶属度是人为给定的，不是计算得出的。

粗糙集理论用在数据库中的知识发现主要体现在：

- 利用等价关系对数据库进行属性约简；
- 利用集合的上、下近似关系获取分类规则。

#### 1. 基本定义

（1）信息表定义

信息表 $S=(U,R,V,f)$ 的定义为：

$U$：是一个非空有限对象（元组）集合，$U=\{x_1,x_2,\cdots,x_n\}$，其中 $x_i$ 为对象（元组）。

$R$：是对象的属性集合，分为两个不相交的子集，即条件属性 $C$ 和决策属性 $D$，$R=C\cup D$。

$V$：是属性值的集合，$V_a$ 是属性 $a\in R$ 的值域。

$f$：是 $U\times R\to V$ 的一个信息函数，它为每个对象 $x$ 的每个属性 $a$ 赋予一个属性值，即 $a\in R$，$x\in U$，$f_a(x)\in V_a$。

（2）等价关系定义

对于 $\forall a\in A$（$A$ 中包含一个或多个属性），$A\subset R$，$x\in U$，$y\in U$，它们的属性值相同，即：

$$f_a(x) = f_a(y) \tag{8.1}$$

成立，称对象 $x$ 和 $y$ 是对属性 $A$ 的等价关系，表示为

$$\mathrm{IND}(A) = \{(x,y) \mid (x,y) \in U \times U, \forall a \in A, f_a(x) = f_a(y)\} \tag{8.2}$$

(3) 等价类定义

在 $U$ 中，对属性集 $A$ 中具有相同等价关系的元素集合称为等价关系 IND(A)的等价类，表示为

$$[x]_A = \{y \mid (x,y) \in \mathrm{IND}(A)\} \tag{8.3}$$

(4) 划分的定义

在 $U$ 中对属性 $A$ 的所有等价类形成的划分表示为

$$A = \{E_i \mid E_i = [x_i]_A, i = 1,2,\cdots\} \tag{8.4}$$

具有特性：

① $E_i \neq \varnothing$

② 当 $i \neq j$ 时，$E_i \cap E_j = \varnothing$

③ $U = \bigcup E_i$

**例 1**：设 $U=\{a$(体温正常)，$b$(体温正常)，$c$(体温正常)，$d$(体温高)，$e$(体温高)，$f$(体温很高)$\}$

对于属性 $A$(体温)的等价关系有：

$$\mathrm{IND}(A) = \{(a,b),(a,c),(b,c),(d,e),(e,d),(a,a),(b,b),(c,c),(d,d),(e,e),(f,f)\}$$

属性 $A$ 的等价类有：

$$E_1 = [a]_A = [b]_A = [c]_A = \{a,b,c\}$$
$$E_2 = [d]_A = [e]_A = \{d,e\}$$
$$E_3 = [f]_A = \{f\}$$

$U$ 中对属性 $A$ 的划分为

$$A = \{E_1,E_2,E_3\} = \{\{a,b,c\},\{d,e\},\{f\}\}$$

### 2. 集合 X 的上、下近似关系

(1) 下近似定义

对任意一个子集 $X \subseteq U$，属性 $A$ 的等价类 $E_i = [x]_A$，有：

$$A_-(X) = \bigcup \{E_i \mid E_i \in A \wedge E_i \subseteq X\} \tag{8.5}$$

或

$$A_-(X) = \{x \mid [x]_A \subseteq X\} \tag{8.6}$$

表示等价类 $E_i = [x]_A$ 中的元素 $x$ 都属于 $X$，即 $\forall x \in A_-(X)$，则 $x$ 一定属于 $X$。

(2) 上近似定义

对任意一个子集 $X \subseteq U$，属性 $A$ 的等价类 $E_i = [x]_A$，有：

$$A^-(X) = U\{E_i \mid E_i \in A \wedge E_i \cap X \neq \varnothing\} \tag{8.7}$$

或

$$A^-(X) = \{x \mid [x]_A \cap X \neq \varnothing\} \tag{8.8}$$

表示等价类 $E_i=[x]_A$ 中的元素 $x$ 可能属于 $X$，即 $\forall x\in A^-(X)$，则 $x$ 可能属于 $X$，也可能不属于 $X$。

(3) 正域，负域和边界的定义

全集 $U$ 可以划分为三个不相交的区域，即正域(Pos)，负域(NEG)和边界(BND)：

$$\mathrm{Pos}_A(X)=A_-(X) \tag{8.9}$$

$$\mathrm{NEG}_A(X)=U-A^-(X) \tag{8.10}$$

$$\mathrm{BND}_A(X)=A^-(X)-A_-(X) \tag{8.11}$$

从上式可见：

$$A^-(X)=A_-(X)+\mathrm{BND}_A(X) \tag{8.12}$$

用图 8.1 说明正域、负域和边界，每一个小长方形表示一个等价类。

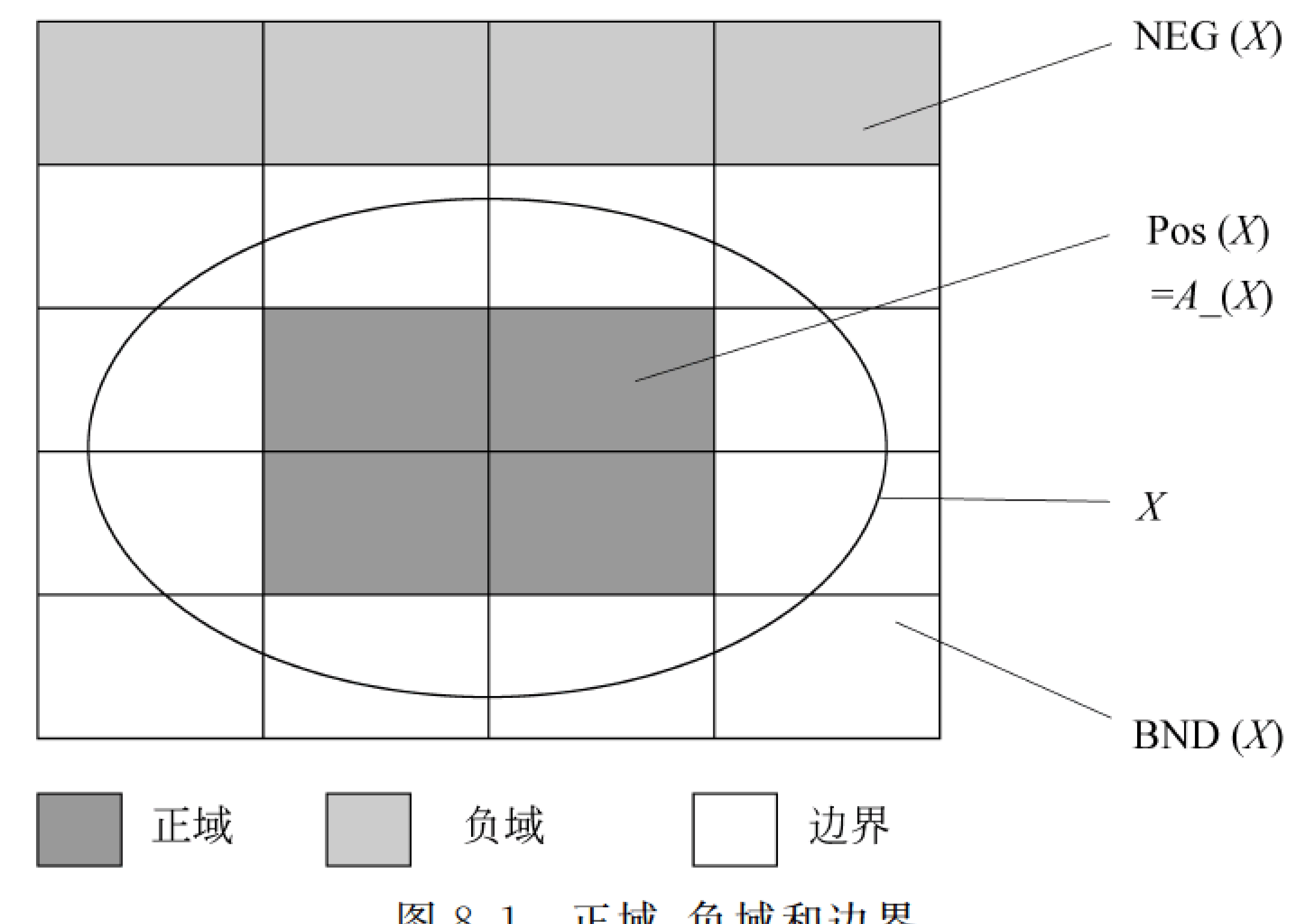



图 8.1 正域、负域和边界

从图 8.1 中可以看出，任意一个元素 $x\in\mathrm{Pos}(X)$，它一定属于 $X$；任意一个元素 $x\in\mathrm{NEG}(X)$，它一定不属于 $X$；集合 $X$ 的上近似是其正域和边界的并集，即

$$A^-(X)=\mathrm{Pos}_A(X)\cup\mathrm{BND}_A(X) \tag{8.13}$$

对于元素 $x\in\mathrm{BND}(X)$，是无法确定其是否属于 $X$ 的，因此对任意元素 $x\in A^-(X)$，只知道 $x$ 可能属于 $X$。

(4) 粗糙集定义

若 $A^-(X)=A_-(X)$，即 $\mathrm{BND}(X)=\varnothing$，即边界为空，称 $X$ 为 $A$ 的可定义集；否则 $X$ 为 $A$ 不可定义的，即 $A^-(X)\neq A_-(X)$，此时称 $X$ 为 $A$ 的 Rough 集(粗糙集)。

(5) 确定度定义

$$\alpha_A(X)=\frac{|U|-|A^-X-A_-X|}{|U|} \tag{8.14}$$

其中 $|U|$ 和 $|A^-X-A_-X|$ 分别表示集合 $U$、$(A^-X-A_-X)$ 中的元素个数。

$\alpha_A(X)$ 的值反映了 $U$ 中能够根据 $A$ 中各属性的属性值就能确定其属于或不属于 $X$ 的比例，也即对 $U$ 中的任意一个对象，根据 $A$ 中各属性的属性值确定它属于或不属于 $X$ 的可信度。

确定度性质：

$$0 \leqslant \alpha_A(X) \leqslant 1 \tag{8.15}$$

(1) 当 $\alpha_A(X)=1$ 时，$U$ 中的全部对象能够根据 $A$ 中各属性的属性值就可以确定其是否属于 $X$，$X$ 为 $A$ 的可定义集。

(2) 当 $0<\alpha_A(X)<1$ 时，$U$ 中的部分对象根据 $A$ 中各属性的属性值可以确定其是否属于 $X$，而另一部分对象是不能确定其是否属于 $X$。$X$ 为 $A$ 的部分可定义集。

(3) 当 $\alpha_A(X)=0$ 时，$U$ 中的全部对象都不能根据 $A$ 中各属性的属性值确定其是否属于 $X$，$X$ 为 $A$ 的完全不可定义集。

当 $X$ 为 $A$ 的部分可定义集或 $X$ 为 $A$ 的完全不可定义集时，称 $X$ 为 $A$ 的 Rough 集（粗糙集）。

**例 2**：对例 1 的等价关系 $A$ 有集合 $X=\{b,c,f\}$ 是粗糙集，计算集合 $X$ 的下近似、上近似、正域、负域和边界。

$U$ 中关于 $A$ 的划分为：

$$A = \{\{a,b,c\},\{d,e\},\{f\}\}$$

有：

$$X \cap \{a,b,c\} = \{b,c\} \neq \varnothing$$
$$X \cap \{d,e\} = \varnothing$$
$$X \cap \{f\} = \{f\} \neq \varnothing$$

可知有：

$$A_-(X) = \{f\}$$
$$A^-(X) = \{a,b,c\} \cup \{f\} = \{a,b,c,f\}$$
$$\mathrm{Pos}_A(X) = A_-(X) = \{f\}$$
$$\mathrm{NEG}_A(X) = U - A^-(X) = \{d,e\}$$
$$\mathrm{BND}_A(X) = A^-(X) - A_-(X) = \{a,b,c\}$$

### 8.1.2 属性约简的粗糙集理论

#### 1. 属性约简概念

在信息表中根据等价关系，可以用等价类中的一个对象（元组）来代表整个等价类，这实际上是按纵方向约简了信息表中数据。对信息表中的数据按横方向进行约简就是看信息表中有无冗余的属性，即去除这些属性后能保持等价性，从而有相同的集合近似，使对象分类能力不会下降。约简后的属性集称为属性约简集，约简集通常不唯一，找到一个信息表的所有约简集不是一个在多项式时间里所解决的问题，求最小约简集（含属性个数最少的约简集）同样是一个困难问题，实际上它是一个 NP-hard 问题。因此研究者提出了很多启发式算法，如基于遗传算法的方法等。

(1) 约简定义

给定一个信息表 $\mathrm{IT}(U,A)$，若有属性集 $B \subseteq A$，且满足 $\mathrm{IND}(B)=\mathrm{IND}(A)$，称 $B$ 为 $A$ 的一个约简，记为 $\mathrm{red}(A)$，即：

$$B = \mathrm{red}(A) \tag{8.16}$$

(2) 核定义

属性集 $A$ 的所有约简的交集称为 $A$ 的核。记为

$$\mathrm{core}(A) = \bigcap \mathrm{red}(A) \tag{8.17}$$

$\mathrm{core}(A)$是 $A$ 中为保证信息表中对象可精确定义的必要属性组成的集合,为 $A$ 中不能约简的重要属性,它是进行属性约简的基础。

上面的约简定义没有考虑决策属性,现研究条件属性 $C$ 相对决策属性 $D$ 的约简。

(3) 正域定义

设决策属性 $D$ 的划分 $A=\{y_1,y_2,\cdots,y_n\}$,条件属性 $C$ 相对于决策属性 $D$ 的正域定义为

$$\mathrm{Pos}_C(D) = \bigcup C_(y_j) \tag{8.18}$$

(4) 条件属性 $C$ 相对于决策属性 $D$ 的约简定义

若 $c\in C$,如果 $\mathrm{Pos}_{(C-\{c\})}(D)=\mathrm{Pos}_C(D)$,则称 $c$ 是 $C$ 中相对于 $D$ 不必要的,即可约简的。否则称 $c$ 是 $C$ 中相对于 $D$ 必要的。

(5) 条件属性 $C$ 相对于决策属性 $D$ 的核定义

若 $R\subseteq C$,如果 $R$ 中每一个 $c\in R$ 都是相对于 $D$ 必要的,则称 $R$ 是相对于 $D$ 独立的。如果 $R$ 是相对于 $D$ 独立的,且 $\mathrm{Pos}_R(D)=\mathrm{Pos}_C(D)$,则称 $R$ 是 $C$ 中相对于 $D$ 的约简,记为 $\mathrm{red}_D(C)$,所有这样简约的交,称为 $C$ 的 $D$ 核,记为

$$\mathrm{core}_D(C) = \bigcap \mathrm{red}_D(C) \tag{8.19}$$

一般情况下,信息系统的属性约简集有多个,但约简集中属性个数最少的最有意义。

### 2. 属性约简实例

气候信息表是 4 个条件属性(天气 $a_1$,温度 $a_2$,湿度 $a_3$,风 $a_4$)和 1 个决策属性(类别 $d$),见表 8.1。

**表 8.1 气候信息表**

| 序号 | 天气 $a_1$ | 气温 $a_2$ | 湿度 $a_3$ | 风 $a_4$ | 类别 $d$ |
|---|---|---|---|---|---|
| 1 | 晴 | 热 | 高 | 无风 | $N$ |
| 2 | 晴 | 热 | 高 | 有风 | $N$ |
| 3 | 多云 | 热 | 高 | 无风 | $P$ |
| 4 | 雨 | 适中 | 高 | 无风 | $P$ |
| 5 | 雨 | 冷 | 正常 | 无风 | $P$ |
| 6 | 雨 | 冷 | 正常 | 有风 | $N$ |
| 7 | 多云 | 冷 | 正常 | 有风 | $P$ |
| 8 | 晴 | 适中 | 高 | 无风 | $N$ |
| 9 | 晴 | 冷 | 正常 | 无风 | $P$ |

续表

| 序号 | 天气 $a_1$ | 气温 $a_2$ | 湿度 $a_3$ | 风 $a_4$ | 类别 $d$ |
|---|---|---|---|---|---|
| 10 | 雨 | 适中 | 正常 | 无风 | $P$ |
| 11 | 晴 | 适中 | 正常 | 有风 | $P$ |
| 12 | 多云 | 适中 | 高 | 有风 | $P$ |
| 13 | 多云 | 热 | 正常 | 无风 | $P$ |
| 14 | 雨 | 适中 | 高 | 有风 | $N$ |

令 $C=\{a_1,a_2,a_3,a_4\}, D=\{d\}$

$\text{IND}(C)=\{\{1\},\{2\},\{3\},\{4\},\{5\},\{6\},\{7\},\{8\},\{9\},\{10\},\{11\},\{12\},\{13\},\{14\}\}$

$\text{IND}(D)=\{\{1,2,6,8,14\},\{3,4,5,7,9,10,11,12,13\}\}$

$\text{Pos}_C(D)=U$

(1) 计算缺少一个属性的等价关系

$\text{IND}(C\backslash\{a_1\})=\{\{1,3\},\{2\},\{4,8\},\{5,9\},\{6,7\},\{10\},\{11\},\{12,14\},\{13\}\}$

$\text{IND}(C\backslash\{a_2\})=\{\{1,8\},\{2\},\{3\},\{4\},\{5,10\},\{6\},\{7\},\{9\},\{11\},\{12\},\{13\},\{14\}\}$

$\text{IND}(C\backslash\{a_3\})=\{\{1\},\{2\},\{3,13\},\{4,10\},\{5\},\{6\},\{7\},\{8\},\{9\},\{11\},\{12\},\{13\},\{14\}\}$

$\text{IND}(C\backslash\{a_4\})=\{\{1,2\},\{3\},\{4,14\},\{5,6\},\{7\},\{8\},\{9\},\{10\},\{11\},\{12\},\{13\}\}$

计算减少一个条件属性相对决策属性的正域

$$\text{Pos}_{(C\backslash\{a_1\})}(D)=\{2,5,9,10,11\}\neq U$$

$$\text{Pos}_{(C\backslash\{a_2\})}(D)=U=\text{Pos}_c(D)$$

$$\text{Pos}_{(C\backslash\{a_3\})}(D)=U=\text{Pos}_c(D)$$

$$\text{Pos}_{(C\backslash\{a_4\})}(D)=\{1,2,3,7,8,9,10,11,12,13\}\neq U$$

由此可知，属性 $a_2$、$a_3$ 是相对于决策属性 $d$ 可省略的，但不一定可以同时省略，而属性 $a_1$ 和 $a_4$ 是相对于决策属性不可省略的，因此：

$$\text{core}(c)=\{a_1,a_4\}$$

(2) 计算同时减少$\{a_2,a_3\}$的等价关系和正域

$\text{IND}(C\backslash\{a_2,a_3\})=\{\{1,8,9\},\{2,11\},\{3,13\},\{4,5,10\},\{6,14\},\{7,12\}\}$

$\text{Pos}_{(C\backslash\{a_2,a_3\})}(D)=\{3,4,5,6,7,10,12,13,14\}\neq U$

说明$\{a_2,a_3\}$同时是不可省略的。

(3) 在$\{a_2,a_3\}$中只能删除一个属性

即存在两个约简：

$$\text{red}_D(C)=\{\{a_1,a_2,a_4\},\{a_1,a_3,a_4\}\}$$

从实例计算中可以看出，信息表的属性约简是在保持条件属性相对决策属性的分类能力不变的条件下，删除不必要的或不重要的属性。一般来讲，条件属性对于决策属性的相对约简不是唯一的，即可能存在多个相对约简。

### 8.1.3 属性约简的粗糙集方法

#### 1. 属性依赖度

(1) 属性依赖度定义

信息表中条件属性 $C$ 和决策属性 $D$，属性 $D$ 依赖属性 $C$ 的依赖度定义为

$$\gamma(C,D) = |\mathrm{Pos}_C(D)| / |U| \tag{8.20}$$

其中 $|\mathrm{Pos}_C(D)|$ 表示正域 $\mathrm{Pos}_C(D)$ 的元素个数，$|U|$ 表示整个对象集合的个数。

$\gamma(C,D)$ 的性质：

① 若 $\gamma=1$，意味着 $\mathrm{IND}(C)\subseteq\mathrm{IND}(D)$，即在已知条件 $C$ 下，可将 $U$ 上全部个体准确分类到决策属性 $D$ 的类别中去，即 $D$ 完全依赖于 $C$。

② 若 $0<\gamma<1$，则称 $D$ 部分依赖于 $C$（$D$ Rough 依赖于 $C$），即在已知条件 $C$ 下，只能将 $U$ 上那些属于正域的个体分类到决策属性 $D$ 的类别中去。

③ 若 $\gamma=0$，则称 $D$ 完全不依赖于 $C$，即利用条件 $C$ 不能分类到 $D$ 的类别中去。

(2) 相关命题

根据属性依赖度定义，可以得到如下命题：

命题 1：如果依赖度 $\gamma=1$，则信息表是一致的，否则是不一致的。

命题 2：每个信息表都能唯一地分解成一个一致信息表（$\gamma=1$）和一个完全不一致信息表（$\gamma=0$）。

#### 2. 属性重要度

(1) 属性重要度的定义

$C,D\subset A$，$C$ 为条件属性集，$D$ 为决策属性集，$a\in C$，属性 $a$ 关于 $D$ 的重要度定义为

$$\mathrm{SGF}(a,C,D) = \gamma(C,D) - \gamma(C-\{a\},D) \tag{8.21}$$

其中 $\gamma(C-\{a\},D)$ 表示在 $C$ 中缺少属性 $a$ 后，条件属性与决策属性的依赖程度。$\mathrm{SGF}(a,C,D)$ 表示 $C$ 中缺少属性 $a$ 后，导致不能被准确分类的对象在系统中所占的比例。

(2) $\mathrm{SGF}(a,C,D)$ 性质

① $\mathrm{SGF}(a,C,D)\in[0,1]$。

② 若 $\mathrm{SGF}(a,C,D)=0$，表示属性 $a$ 关于 $D$ 是可省的。因为从属性集中去除属性 $a$ 后，$C-\{a\}$ 中的信息，原来可被准确分类所有对象仍能准确划分到各决策类中去。

③ $\mathrm{SGF}(a,C,D)\neq0$，表示属性 $a$ 关于 $D$ 是不可省的。因为从属性集 $C$ 中去除属性 $a$ 后，某些原来可被准确分类的对象不再能被准确划分。

#### 3. 最小属性集概念

对信息系统最广泛的应用是数据库。在数据库中根据决策属性将一组对象划分为各不相交的等价集（决策类），希望能通过条件属性来决定每一个决策类，并产生每一个类的判定规则。大多数情况下，对每个给定的学习任务，数据库中存在一些不重要属性，希望找到一个最小的相关属性集，它具有与全部条件属性同样的区分决策属性所划分的决策类的能力，

从最小属性集中产生的规则会更简练和更有意义。

最小属性集定义：设 $C$、$D$ 分别是信息系统 $S$ 的条件属性集和决策属性集，属性集 $P(P\subseteq C)$是 $C$ 的一个最小属性集，当且仅当 $\gamma(P,D)=\gamma(C,D)$并且 $\forall P'\subset P,\gamma(P',D)\neq\gamma(P,D)$，说明若 $P$ 是 $C$ 的最小属性集，则 $P$ 具有与 $C$ 同样的区分决策类的能力。

需要注意的是，$C$ 的最小属性集一般是不唯一的，而要找到所有的最小属性集是一个 NP 问题。在大多数应用中，没有必要找到所有的最小属性集。用户可以根据不同的原则来选择一个他认为最好的最小属性集。比如，选择具有最少属性个数的最小属性集。

### 8.1.4 粗糙集方法的规则获取

通过分析 $U$ 中的两个划分 $C=\{E_i\}$和 $D=\{Y_j\}$之间的关系，把 $C$ 视为分类条件，把 $D$ 视为分类结论，可以得到下面的分类规则：

(1) 当 $E_i\cap Y_j\neq\varnothing$时，则有：

$$r_{ij}:\mathrm{Des}(E_i)\rightarrow\mathrm{Des}(Y_j) \tag{8.22}$$

$\mathrm{Des}(E_i)$和 $\mathrm{Des}(Y_j)$分别是等价集 $E_i$ 和等价集 $Y_j$ 中的特征描述。

① 当 $E_i\cap Y_j=E_i$($E_i$ 完全被 $Y_j$ 包含)即下近似时，建立的规则 $r_{ij}$ 是确定的，规则的可信度 cf=1.0。

② 当 $E_i\cap Y_j\neq E_i$($E_i$ 部分被 $Y_j$ 包含)即上近似时，建立的规则 $r_{ij}$ 是不确定的，规则的可信度为

$$\mathrm{cf}=\frac{|E_i\cap Y_j|}{|E_i|} \tag{8.23}$$

(2) 当 $E_i\cap Y_j=\varnothing$时($E_i$ 不被 $Y_j$ 包含)，$E_i$ 和 $Y_j$ 不能建立规则。

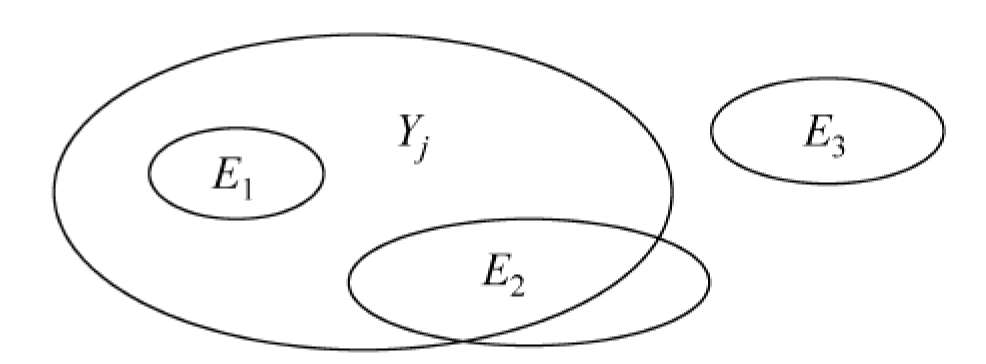


图 8.2 $E_i$ 和 $Y_j$ 的上、下近似关系

### 8.1.5 粗糙集方法的应用实例

通过实例说明属性约简和规则获取方法。有表 8.2 所示的数据。

**表 8.2 流感实例数据**

| | C(条件属性) | | | D(决策属性) |
|---|---|---|---|---|
| $U$ | 头痛($a$) | 肌肉痛($b$) | 体温($c$) | 流感($d$) |
| $e_1$ | 是(1) | 是(1) | 正常(0) | 否(0) |
| $e_2$ | 是(1) | 是(1) | 高(1) | 是(1) |
| $e_3$ | 是(1) | 是(1) | 很高(2) | 是(1) |
| $e_4$ | 否(0) | 是(1) | 正常(0) | 否(0) |
| $e_5$ | 否(0) | 否(0) | 高(1) | 否(0) |
| $e_6$ | 否(0) | 是(1) | 很高(2) | 是(1) |
| $e_7$ | 是(1) | 否(0) | 高(1) | 是(1) |

**1. 等价集下近似和依赖度的计算**

(1) 条件属性 $C(a,b,c)$的等价集

由于各元组(对象)之间不存在等价关系,每个元组组成一个等价集,共七个:$E_1\{e_1\}$,$E_2\{e_2\}$,$E_3\{e_3\}$,$E_4\{e_4\}$,$E_5\{e_5\}$,$E_6\{e_6\}$,$E_7\{e_7\}$。

(2) 决策属性 $D(d)$的等价集

按属性取值,共有两个等价集:$Y_1$:$\{e_1,e_4,e_5\}$;$Y_2$:$\{e_2,e_3,e_6,e_7\}$。

(3) 决策属性的各等价集的下近似集为

$$C_Y_1 = \{E_1,E_4,E_5\} = \{e_1,e_4,e_5\}$$
$$C_Y_2 = \{E_2,E_3,E_6,E_7\} = \{e_2,e_3,e_6,e_7\}$$

此例不存在上近似集。

(4) 计算 $\text{Pos}(C,D)$和 $\gamma(C,D)$

$$\text{Pos}(C,D) = C_Y_1 \cup C_Y_2 = \{e_1,e_2,e_3,e_4,e_5,e_6,e_7\}$$
$$|\text{Pos}(C,D)| = 7,\ |U| = 7, \gamma(C,D) = 1$$

**2. 各属性重要度计算**

(1) $a$ 的重要度计算

- 条件属性 $C(b,c)$的等价集。

$$E_1\{e_1,e_4\},E_2\{e_2\},E_3\{e_3,e_6\},E_4\{e_5,e_7\}$$

- 决策属性 $D(d\ )$的等价集(同上)。
- 决策属性各等价集的下近似集。

$$C_Y_1 = \{E_1\} = \{e_1,e_4\}$$
$$C_Y_2 = \{E_2,E_3\} = \{e_2,e_3,e_6\}$$

- 计算 $\text{Pos}(C-\{a\},D)$和 $\gamma(C-\{a\},D)$。

$$\text{Pos}(C-\{a\},D) = C_Y_1 \cup C_Y_2 = \{e_1,e_2,e_3,e_4,e_6\}$$
$$|\text{Pos}(C-\{a\},D)| = 5$$
$$\gamma(C-\{a\},D) = 5/7$$

- 属性 $a$ 的重要程度。

$$\text{SGF}(C-\{a\},D) = \gamma(C,D) - \gamma(C-\{a\},D) = 2/7 \neq 0$$

- 结论:属性 $a$ 是不可省略的。

(2) $b$ 的重要度计算

- 条件属性 $C(a,c)$的等价集。

  去掉属性 $b$ 后,元组中只出现 $e_2$ 和 $e_7$ 的等价,其他元组均不等价,等价集共 6 个:$E_1\{e_1\}$,$E_2\{e_2,e_7\}$,$E_3\{e_3\}$,$E_4\{e_4\}$,$E_5\{e_5\}$,$E_6\{e_6\}$。
- 决策属性 $D(d)$的等价集(同上)。
- 决策属性的各等价集的下近似集。

$$C_Y_1 = \{E_1,\ E_4,E_5\} = (e_1,e_4,e_5)$$
$$C_Y_2 = \{E_2,\ E_3,\ E_6\} = (e_2,e_7,e_3,e_6)$$

• 计算 $\text{Pos}(C-\{b\},D)$。

$$\text{Pos}(C-\{b\},D)=C_Y_1\cup C_Y_2=(e_1,e_2,e_3,e_4,e_5,e_6,e_7)$$

$$|\text{Pos}(C-\{b\},D)|=7,\gamma(C-\{b\},D)=1$$

• 属性 $b$ 的重要度。

$$\text{SGF}(C-\{b\},D)=\gamma(C,D)-\gamma(C-\{b\},D)=0$$

• 结论：属性 $b$ 是可省略的。

**3. 简化数据表**

在原数据表中删除肌肉痛($b$)属性后，元组 $e_7$ 和 $e_2$ 相同，合并成表 8.3 所示的简化数据表。

**表 8.3　流感数据简化表**

| $U$ | 头痛($a$) | 体温($c$) | 流感($d$) |
|---|---|---|---|
| $e_1'$ | 是(1) | 正常(0) | 否(0) |
| $e_2'$ | 是(1) | 高(1) | 是(1) |
| $e_3'$ | 是(1) | 很高(2) | 是(1) |
| $e_4'$ | 否(0) | 正常(0) | 否(0) |
| $e_5'$ | 否(0) | 高(1) | 否(0) |
| $e_6'$ | 否(0) | 很高(2) | 是(1) |

**4. 等价集、上下近似集的计算**

(1) 条件属性的等价集

由于各元组之间不存在等价关系，故有 6 个等价集：$E_1'\{e_1'\}$；$E_2'\{e_2'\}$；$E_3'\{e_3'\}$；$E_4'\{e_4'\}$；$E_5'\{e_5'\}$；$E_6'\{e_6'\}$。

(2) 决策属性 $D(d)$ 的等价集

按属性取值，共有两个等价集：$Y_1'\{e_1',e_4',e_5'\}$；$Y_2'\{e_2',e_3',e_6'\}$。

**5. 获取规则**

(1) 如图 8.3 所示，由于 $E_1'\cap Y_1'=E_1'$，$E_4'\cap Y_1'=E_4'$，$E_5'\cap Y_1'=E_5'$，故有规则

$r_{11}$：$\text{Des}(E_1')\rightarrow\text{Des}(Y_1')$，即

$$a=1\wedge c=0\rightarrow d=0,\quad \text{cf}=1$$

$r_{41}$：$\text{Des}(E_4')\rightarrow\text{Des}(Y_1')$，即

$$a=0\wedge c=0\rightarrow d=0,\quad \text{cf}=1$$

$r_{51}$：$\text{Des}(E_5')\rightarrow\text{Des}(Y_1')$，即

$$a=0\wedge c=1\rightarrow d=0,\quad \text{cf}=1$$

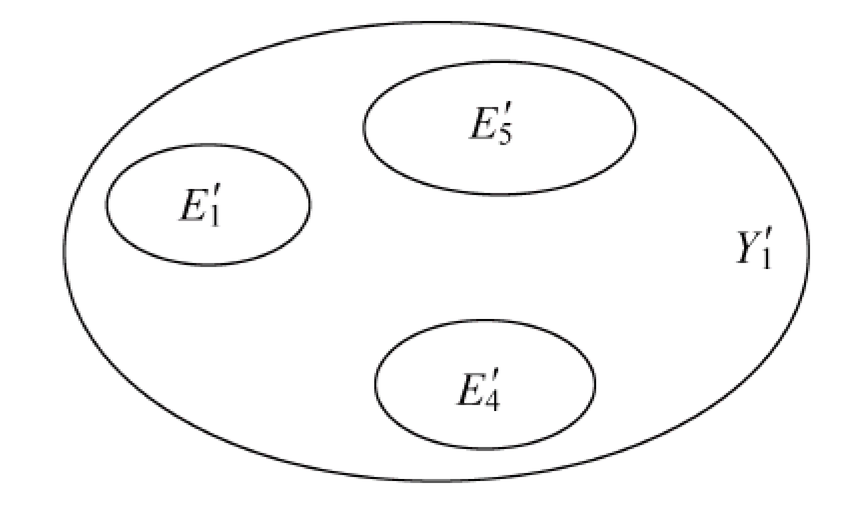


图 8.3　$Y_1'$ 与 $E_1'$、$E_4'$、$E_5'$ 最小包含图

(2) 由于 $E_2'\cap Y_2'=E_2'$，$E_3'\cap Y_2'=E_3'$，$E_6'\cap Y_2'=E_6'$，有规则

$r_{22}$：$\mathrm{Des}(E_2')\to\mathrm{Des}(Y_2')$，即 $a=1\wedge c=1\to d=1,\mathrm{cf}=1$。

$r_{32}$：$\mathrm{Des}(E_3')\to\mathrm{Des}(Y_2')$，即 $a=1\wedge c=2\to d=1,\mathrm{cf}=1$。

$r_{62}$：$\mathrm{Des}(E_6')\to\mathrm{Des}(Y_2')$，即 $a=0\wedge c=2\to d=1,\mathrm{cf}=1$。

#### 6. 规则化简

(1) 对 $r_{11}$ 和 $r_{41}$ 进行合并，有：

$$(a=0\vee a=1)\wedge c=0\to d=0$$

其中 $a$ 的取值包括了他的全部取值，故属性 $a$ 可删除，即：

$$c=0\to d=0$$

(2) 对 $r_{32}$ 和 $r_{62}$ 进行合并，有：

$$(a=1\vee a=0)\wedge c=2\to d=1$$

同样，可删除属性 $a$，得到：

$$c=2\to d=1$$

#### 7. 最后的规则

(1) 体温＝正常→流感＝否(即 $c=0\to d=0$)

(2) 头痛＝否∧体温＝高→流感＝否(即 $a=0\wedge c=1\to d=0$)

(3) 体温＝很高→流感＝是(即 $c=2\to d=1$)

(4) 头痛＝是∧体温＝高→流感＝是(即 $a=1\wedge c=1\to d=1$)

## 8.2 K-均值聚类

### 8.2.1 聚类方法简介

#### 1. 聚类方法

聚类(Cluster)问题描述为：给定数据集合 $D$，把它划分成一组聚类：$\{C_1,C_2,\cdots,C_k\}$，$C_i\in D$，使得不同类中的数据尽可能的不相似(或距离较远)，而同一类中的数据尽可能的相似(或距离较近)。如果 $k=1$ 或 $k=|D|$（$|D|$表示集合 $D$ 的元素个数)，则称为平凡聚类。

按照聚类结果来划分聚类算法，分为三种：

(1) 覆盖(Coverage)：如果每个对象至少属于一个聚类，则称聚类为覆盖的，否则为非覆盖的；

(2) 相交(Separation)：如果至少一个对象属于一个以上的聚类，则称聚类为模糊的，反之，如果任意两个聚类的交集为空，则称聚类是确定的；

(3) 结构(Structure)：如果两个聚类或者不相交或者其中一个是另一个的子集，则称聚类为层次的，否则为非层次的。

按照聚类的原理和方法来划分聚类算法，也分为三种：

(1) 层次聚类

层次聚类(Hierarchical Clustering)方法递归地对对象进行合并或者分裂，直到满足某

一终止条件。层次聚类的结果可以用二叉树表示，树中的每个结点都是一个聚类，下层的聚类是上层聚类的嵌套，每一层结点构成一组划分。根据二叉树生成的顺序，可以把层次聚类方法分为合并型层次聚类和分解型层次聚类两种。

合并型层次聚类从单成员聚类开始，把它们逐渐合并成更大的聚类，在每一层中，相距最近的两个聚类被合并。相反，分解型层次聚类从包含所有对象的一个聚类开始，把它逐渐分解成更小的聚类。

(2) 划分聚类(Partitional Clustering)

给定聚类数目 $k$ 和目标函数 $F$，划分聚类算法把 $D$ 划分成 $k$ 个类，使得目标函数在此划分下达到最优。划分算法把聚类问题转化成一个组合优化问题，从一个初始划分或者一个初始聚点集合开始，利用迭代控制策略优化目标函数。

最常用的目标函数是：$\sum \min d(x_i, m_j)$，其中 $m_j$ 是 $C_j$ 的中心(k-means 算法)或者是 $C_j$ 中离中心最近的一个对象(k-medoids 算法)。

K-means(K-均值)算法是最流行的聚类算法之一。它首先随机地选取 $k$ 个初始聚类中心，并把每个对象分配给离它最近的中心，从而得到一个初始聚类。然后，计算出当前每个聚类的重心作为新的聚类中心，并把每个对象重新分配到最近的中心。如果新的聚类的质量优于原先的聚类，则用新聚类代替原聚类。循环执行这一过程直至聚类质量不再提高为止。后来，许多变形算法都是在基本 k-means 算法的基础上做了改进。

(3) 基于密度的聚类

以空间中的一点为中心，单位体积内点的个数称为该点的密度，从直观来看，聚类的内部点的密度较大，而聚类之间点的密度较小。基于密度的聚类(Density-based Clustering)根据空间密度的差别，把具有相似密度的点作为聚类。由于密度是一个局部概念，因此这类算法又称为局部聚类(Local Clustering)。基于密度的聚类通常只扫描一次数据库，所以又称为单次扫描聚类(Single Scan Clustering)。

对于空间中的一个对象，如果它在给定半径 Eps 的邻域中的对象个数大于某个给定数值 Minpts，则该对象被称为核心对象(core point)，否则称为边界对象。由一个核心对象密度可达的所有对象构成一个聚类。

层次聚类和划分聚类是最常用的聚类方法。

### 2. 相似度量方法

对象间的距离或相似度是聚类的核心，常常按照对象之间的相似性进行划分，划分的结果使某种表示聚类质量的评价函数最优。数据的类型不同，相似性的含义也不同。例如在数值型数据库中，两个对象的相似度是指它们在几何空间中互相邻近的程度；在分类型数据库中，两个对象的相似性是指它们在同一个属性上取值相同；在交易型数据库中，两个交易相似是指它们包含相同的数据项。

聚类可以分为两类：对对象聚类称为 Q 型聚类，往往用距离或相似系数来度量相似性；对属性聚类称为 R 型聚类时，常根据相关系数或关联系数来度量相似性。

(1) 对象的距离

假设每个对象有 $m$ 个属性，可以把一个对象视为 $m$ 维空间的一个点，$n$ 个对象就是 $m$

维空间中的 $n$ 个点。从直观上看，属于同一类的对象在空间中应该互相靠近，而不同类的对象之间的距离要大得多，很自然地想到用它们之间的距离来衡量它们之间的相似程度。距离越小，对象间的相似性越大。

在聚类分析中，常用的距离公式有：

- 明考夫斯基(Minkowski)距离：

$$d_{ij} = \left( \sum_{k=1}^{m} | x_{ik} - x_{jk} |^{p} \right)^{\frac{1}{p}}$$

- 曼哈顿(Manhattan)距离：

$$d_{ij} = \sum_{k=1}^{m} | x_{ik} - x_{jk} |$$

- 欧氏(Euclidean)距离：

$$d_{ij} = \left( \sum_{k=1}^{m} | x_{ik} - x_{jk} |^{2} \right)^{\frac{1}{2}}$$

其中最常用的是欧氏距离，对坐标系进行平移和旋转变换之后，欧氏距离保持不变。

(2) 对象的相似系数

相似系数与距离相反，相似系数越大，对象间的相似性越大。$X_i$、$X_j$ 的相似系数 $r_{ij}$ 有：

- 最大最小法：

$$r_{ij} = \frac{\sum_{k=1}^{m} \min(x_{ik}, x_{jk})}{\sum_{k=1}^{m} \max(x_{ik}, x_{jk})}$$

- 算术平均最小法：

$$r_{ij} = \frac{\sum_{k=1}^{m} \min(x_{ik}, x_{jk})}{\frac{1}{2} \sum_{k=1}^{m} (x_{ik} + x_{jk})}$$

- 夹角余弦法：

$$r_{ij} = \frac{\left| \sum_{k=1}^{m} x_{ik} \times x_{jk} \right|}{\sqrt{\left( \sum_{k=1}^{m} x_{ik}^{2} \right) \left( \sum_{k=1}^{m} x_{jk}^{2} \right)}}$$

## 8.2.2 K-均值聚类方法与实例

### 1. K-均值聚类方法

K-均值方法是一种常用的基于划分的聚类方法，它根据最终分类的个数 $k$ 随机地选取 $k$ 个初始的聚类中心，不断地迭代，直到达到目标函数的最小值，即得到最终的聚类结果。其中，目标函数通常采用平方误差准则，即：

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} | p - m_i |^{2}$$

其中，$E$ 表示所有聚类对象的平方误差的和，$p$ 是聚类对象，$m_i$ 是类 $C_i$ 的各聚类对象（样本）的平均值，即：

$$m_i = \frac{\sum_{p \in C_i} p}{|C_i|}$$

其中，$|C_i|$ 表示类 $C_i$ 的聚类对象的数目。

因为在每一次迭代中，每一个点都要计算和各聚类中心的距离，并将距离最近的类作为该点所属的类，所以 K-均值方法的算法复杂度为 $O(knt)$，其中 $k$ 表示聚类数，$n$ 表示结点数，$t$ 是迭代次数。$k$ 的典型取值是 2～10。

K-均值方法是解决聚类问题的一种经典算法，它是一种爬山式的搜索算法。这种算法简单、快速。然而，K-均值方法对初值敏感，对于不同的初始值，可能会导致不同的聚类结果。此外，K-均值算法是基于梯度下降的算法，由于目标函数局部极小值点的存在，以及算法的贪心性，因此算法可能会陷入局部最优，而无法达到全局最优。

**2. K-均值聚类方法实例**

假设给定如下要进行聚类的元组：

{2，4，10，12，3，20，30，11，25}

并假设 $k=2$。初始时用前两个数值作为类的均值：$m_1=2$ 和 $m_2=4$。利用欧几里得距离，可得 $K_1=\{2,3\}$ 和 $K_2=\{4,10,12,20,30,11,25\}$。数值 3 与两个均值的距离相等，所以任意地选择 $K_1$ 作为其所属的类。再计算两个类的均值可得 $m_1=2.5$ 和 $m_2=16$。重新对类中的成员进行分配可得 $K_1=\{2,3,4\}$和 $K_2=\{10,12,20,30,11,25\}$。不断重复这个过程可得：

| $m_1$ | $m_2$ | $K_1$ | $K_2$ |
|---|---|---|---|
| 3 | 18 | {2，3，4，10} | {12，20，30，11，25} |
| 4.5 | 19.6 | {2，3，4，10，11，12} | {20，30，25} |
| 7 | 25 | {2，3，4，10，11，12} | {20，30，25} |

注意在最后两步中，类的成员是一致的。再往下循环均值不会再改变，因此，该问题的答案为 $K_1=\{2,3,4,10,11,12\}$和 $K_2=\{20,30,25\}$。

虽然 K-均值算法产生的结果通常都不错，但在时间上并非高效，并且不具有很好的可伸缩性。从上一步到下一步的迭代过程中，通过存储距离信息，可以减少一些必须进行的距离计算的实际次数。

## 8.3　关联规则挖掘

关联规则（Association Rule）挖掘是发现大量数据库中项集之间的关联关系。随着大量数据的增加和存储，许多人士对于从数据库中挖掘关联规则越来越感兴趣。从大量商业事务中发现有趣的关联关系，可以帮助许多商业决策的制定，如分类设计、交叉购物等。

目前，关联规则挖掘已经成为数据挖掘领域重要的研究方向。关联规则模式属于描述型模式，发现关联规则的算法属于无监督学习的方法。

Agrawal 等于 1993 年首先提出了挖掘顾客交易数据库中项集间的关联规则问题，以后诸多研究人员对关联规则的挖掘问题进行了大量的研究。他们的工作包括对原有的算法进行优化，如引入随机采样、并行的思想等，以提高算法挖掘规则的效率；对关联规则的应用进行推广。

最近也有独立于 Agrawal 的频繁集方法的工作，以克服频繁集方法的一些缺陷，探索挖掘关联规则的新方法。同时随着 OLAP 技术的成熟和应用，将 OLAP 和关联规则结合也成了一个重要的方向。也有一些工作注重于对挖掘到的模式的价值进行评估，他们提出的模型建议了一些值得考虑的研究方向。

本章主要给出了关联规则挖掘的基本概念、核心挖掘算法。

### 8.3.1 关联规则的挖掘原理

关联规则是发现交易数据库中不同商品(项)之间的联系，这些规则找出顾客购买行为模式，如购买了某一商品对购买其他商品的影响。发现这样的规则可以应用于商品货架设计、货存安排以及根据购买模式对用户进行分类。现实中，这样的例子很多。最典型的例子是超级市场利用前端收款机收集存储了大量的售货数据，这些数据是一条条的购买事务记录，每条记录存储了事务处理时间，顾客购买的物品、物品的数量及金额等。这些数据中常常隐含形式如下的关联规则：

在购买铁锤的顾客当中，有 70%的人同时购买了铁钉。

这些关联规则很有价值，商场管理人员可以根据这些关联规则更好地规划商场，如把铁锤和铁钉这样的商品摆放在一起，就能够促进销售。

有些数据不像售货数据那样很容易就能看出一个事务是许多物品的集合，但稍微转换一下思考角度，仍然可以像售货数据一样处理。比如人寿保险，一份保单就是一个事务。保险公司在接受保险前，往往需要记录投保人详尽的信息，有时还需要投保人到医院做身体检查。保单上记录有投保人的年龄、性别、健康状况、工作单位、工作地址、工资水平等。

这些投保人的个人信息就可以看做事务中的物品。通过分析这些数据，可以得到类似以下这样的关联规则：

年龄在 40 岁以上，工作在 A 区的投保人当中，有 45%的人曾经向保险公司索赔过。在这条规则中，“年龄在 40 岁以上”是物品甲，“工作在 A 区”是物品乙，“向保险公司索赔过”则是物品丙。可以看出，A 区可能污染比较严重，环境比较差，导致工作在该区的人健康状况不好，索赔率也相对比较高。

#### 1. 基本原理

设 $I=\{i_1, i_2, \cdots, i_m\}$是项(Item)的集合。记 $D$ 为事务(Transaction)的集合(事务数据库)，事务 $T$ 是项的集合，并且 $T \subseteq I$。对每一个事务有唯一的标识，如事务号，记作 TID。设 $A$ 是 $I$ 中一个项集，如果 $A \subseteq T$，那么称事务 $T$ 包含 $A$。

**定义 1** 关联规则是形如 $A \rightarrow B$ 的蕴涵式，这里 $A \subset I$，$B \subset I$，并且 $A \cap B = \varnothing$。

**定义 2** 规则的支持度。

规则 $A \rightarrow B$ 在数据库 $D$ 中具有支持度 $S$，表示 $S$ 是 $D$ 中事务同时包含 $AB$ 的百分比，它是概率 $P(AB)$，即：

$$S(A \rightarrow B) = P(AB) = \frac{|AB|}{|D|} \tag{8.24}$$

其中 $|D|$ 表示事务数据库 $D$ 的个数，$|AB|$ 表示 $A$、$B$ 两个项集同时发生的事务个数。

**定义 3** 规则的可信度。

规则 $A \rightarrow B$ 具有可信度 $C$，表示 $C$ 是包含 $A$ 项集的同时也包含 $B$ 项集，相对于包含 $A$ 项集的百分比，这是条件概率 $P(B|A)$，即：

$$C(A \rightarrow B) = P(B \mid A) = \frac{|AB|}{|A|} \tag{8.25}$$

其中 $|A|$ 表示数据库中包含项集 $A$ 的事务个数。

**定义 4** 阈值。

为了在事务数据库中找出有用的关联规则，需要由用户确定两个阈值：最小支持度(min_sup)和最小可信度(min_conf)。

**定义 5** 项的集合称为项集(Itemset)，包含 $k$ 个项的项集称之为 K-项集。如果项集满足最小支持度，则它称为频繁项集(Frequent Itemset)。

**定义 6** 关联规则。

同时满足最小支持度(min_sup)和最小可信度(min_conf)的规则称之为关联规则，即 $S(A \rightarrow B) > \text{min_sup}$ 且 $C(A \rightarrow B) > \text{min_conf}$ 成立时，规则 $A \rightarrow B$ 称为关联规则，也可以称为强关联规则。

### 2. 关联规则挖掘过程

关联规则的挖掘一般分为两个过程：

(1) 找出所有的频繁项集：根据定义，这些项集的支持度应该满足最小支持度。

(2) 由频繁项集产生关联规则：根据定义，这些规则必须满足最小支持度和最小可信度。

在这两步中，第二步是在第一步的基础上进行的，工作量非常小。挖掘关联规则的总体性能由第一步决定。

### 3. 关联规则的兴趣度

关联规则主要是考虑同时购买商品的事务的相关性。对于不购买商品的事务与购买商品的事务的关系的研究，需要引入兴趣度概念。

先通过一个具体的例子说明不购买商品与购买商品的关系。设 $I=$(咖啡，牛奶)，交易集 $D$，经过对 $D$ 的分析，得到如表 8.4 所示的表格。

由表 8.4 可以了解到如果设定 minsupp=0.2，minconf=0.6，按照现有的挖掘算法就可以得到如下关联规则。

$$\text{买牛奶} \rightarrow \text{买咖啡} \quad s = 0.2 \quad c = 0.8 \tag{8.26}$$

表 8.4　交易集的分析

| | 买　咖　啡 | 不 买 咖 啡 | 合　　计 |
| --- | --- | --- | --- |
| 买牛奶 | 20 | 5 | 25 |
| 不买牛奶 | 70 | 5 | 75 |
| 合计 | 90 | 10 | 100 |

即 80%的人买了牛奶就会买咖啡。这一点从逻辑上看是完全合理正确的。

但从表中,我们同时也可以毫不费神的得到结论:90%的人肯定会买咖啡。换句话说,买牛奶这个事件对于买咖啡这个事件的刺激作用(80%)并没有想象中的(90%)那么大。反而是规则

$$\text{买咖啡} \rightarrow \text{不买牛奶} \quad s = 0.7 \quad c = 0.78 \tag{8.27}$$

的支持度和可信度分别为 0.7 和 0.78,更具有商业销售的指导意义。

从上面这个例子中可以发现,目前基于支持度－可信度的关联规则的评估体系存在着问题;同时,现有的挖掘算法只能挖掘出类似于式(8.27)的规则,而对类似式(8.28)的带有类似于“不买牛奶”之类的负属性项的规则却无能为力,而这种知识往往具有更重要的价值。国内外围绕这个问题展开了许多研究。引入兴趣度概念,分析项集 $A$ 与项集 $B$ 的关系程度。

**定义 7**　兴趣度为

$$I(A \rightarrow B) = \frac{P(AB)}{P(A)P(B)} \tag{8.28}$$

公式(8.29)反映了项集 $A$ 与项集 $B$ 的相关程度。若

$$I(A \rightarrow B) = 1, \quad \text{即} \quad P(AB) = P(A)P(B)$$

表示项集 $A$ 出现和项集 $B$ 是相互独立的。若

$$I(A \rightarrow B) < 1$$

表示 $A$ 出现和 $B$ 出现是负相关的。若

$$I(A \rightarrow B) > 1$$

表示 $A$ 出现和 $B$ 出现是正相关的。意味着 $A$ 的出现蕴含 $B$ 的出现。

在兴趣度的使用中,一条规则的兴趣度越大于 1 说明我们对这条规则越感兴趣(即其实际利用价值越大);一条规则的兴趣度越小于 1 说明我们对这条规则的反面规则越感兴趣(即其反面规则的实际利用价值越大);显然,兴趣度 $I$ 不小于 0。

下面从兴趣度的角度来看一下前面那个牛奶与咖啡的例子。我们列出所有可能的规则描述及其对应的支持度、可信度和兴趣度,如表 8.5 所示。

表 8.5　所有可能的关联规则

| | Rules | $S$ | $C$ | $I$ |
| --- | --- | --- | --- | --- |
| 1 | 买牛奶→买咖啡 | 0.2 | 0.8 | 0.89 |
| 2 | 买咖啡→买牛奶 | 0.2 | 0.22 | 0.89 |
| 3 | 买牛奶→不买咖啡 | 0.05 | 0.2 | 2 |

续表

| | Rules | $S$ | $C$ | $I$ |
|---|---|---|---|---|
| 4 | 不买咖啡→买牛奶 | 0.05 | 0.5 | 2 |
| 5 | 不买牛奶→买咖啡 | 0.7 | 0.93 | 1.037 |
| 6 | 买咖啡→不买牛奶 | 0.7 | 0.78 | 1.037 |
| 7 | 不买牛奶→不买咖啡 | 0.05 | 0.067 | 0.67 |
| 8 | 不买咖啡→不买牛奶 | 0.05 | 0.2 | 0.87 |

在此只考虑第1、2、3、6共4条规则。由于$I_1$，$I_2<1$，因此在实际中它的价值不大；$I_3$，$I_6>1$都可以列入进一步考虑的范围。

公式(8.29)等价于

$$I(A \to B) = \frac{P(AB)}{P(A)P(B)} = \frac{P(B \mid A)}{P(B)} \tag{8.29}$$

有人称公式(8.30)为作用度(Lift)，表示关联规则$A \to B$的“提升”。如果作用度(兴趣度)不大于1，则此关联规则就没有意义了。

概括地说：可信度是对关联规则地准确度的衡量。支持度是对关联规则重要性的衡量。支持度说明了这条规则在所有事务中有多大的代表性，显然支持度越大，关联规则越重要。有些关联规则可信度虽然很高，但支持度却很低，说明该关联规则实用的机会很小，因此也不重要。

兴趣度(作用度)描述了项集$A$对项集$B$的影响力的大小。兴趣度(作用度)越大，说明项集$B$受项集$A$的影响越大。

### 8.3.2 Apriori 算法基本思想

Agrawal 等设计了基于频繁集理论的 Apriori 算法。Apriori 是挖掘关联规则的一个重要方法。这是一个基于两阶段频繁集思想的方法，将关联规则挖掘算法的设计分解为两个子问题：

- 找到所有支持度大于最小支持度的项集(Itemset)，这些项集称为频繁项集(Frequent Itemset)。
- 使用第1步找到的频繁项集产生期望的规则。

Apriori 使用一种称作逐层搜索的迭代方法，“K-项集”用于探索“K+1-项集”。

首先，找出频繁“1-项集”的集合。该集合记作$L_1$。$L_1$用于找频繁“2-项集”的集合$L_2$，而$L_2$用于找$L_3$，如此下去，直到不能找到“K-项集”为止。找每个$L_K$需要一次数据库扫描。

#### 1. Apriori 性质

性质：频繁项集的所有非空子集都必须也是频繁的。

该性质表明，如果项集$B$不满足最小支持度阈值 min-sup，则$B$不是频繁的，即$P(B)$ $<$min-sup。如果项$A$添加到$B$，则结果项集(即$B \cup A$)不可能比$B$更频繁地出现。因此，

$B \cup A$ 也不是频繁的，即 $P(B \cup A) <$ min-sup。

Apriori 性质可用于压缩搜索空间。

**2. "K-项集"产生"K+1-项集"**

设 K-项集 $L_K$，K+1-项集 $L_{K+1}$，产生 $L_{K+1}$ 的候选集 $C_{K+1}$。有公式：

$$C_{K+1} = L_K \times L_K = \{X \cup Y, 其中\ X, Y \in L_K, \mid XY \mid = K+1\}$$

其中 $C_1$ 是 1-项集的集合，取自所有事务中的单项元素。

如 $L_1 = \{\{A\}, \{B\}\}$

$C_2 = \{A\} \cup \{B\} = \{A, B\}$，且 $|AB| = 2$

$L_2 = \{\{A, B\}, \{A, C\}\}$

$C_3 = \{A, B\} \cup \{A, C\} = \{A, B, C\}$，且 $|ABC| = 3$

**3. Apriori 算法中候选项集与频繁项集的产生实例**

有表 8.6 所示的事务数据库，Apriori 算法步骤如下：

对于下述例子的事务数据库产生频繁项集。

**表 8.6 事务数据库例**

| 事务 ID | 事务的项目集 | 事务 ID | 事务的项目集 |
|---|---|---|---|
| $T_1$ | $A,B,E$ | $T_6$ | $B,C$ |
| $T_2$ | $B,D$ | $T_7$ | $A,C$ |
| $T_3$ | $B,C$ | $T_8$ | $A,B,C,E$ |
| $T_4$ | $A,B,D$ | $T_9$ | $A,B,C$ |
| $T_5$ | $A,C$ | | |

(1) 在算法的第一次迭代，每个项都是候选 1-项集的集合 $C_1$ 的成员。算法扫描所有的事务，对每个项的出现次数计数，见图 8.4 中第 1 列。

(2) 假定最小事务支持计数为 2(即 min-sup$=2/9=22\%$)，可以确定频繁 1-项集的集合 $L_1$。它由具有最小支持度的候选 1-项集组成，见图 8.4 中第 2 列。

(3) 为发现频繁 2-项集的集合 $L_2$，算法使用 $L_1 \times L_1$ 来产生候选集 $C_2$，见图 8.4 中第 3 列。

(4) 扫描 $D$ 中事务，计算 $C_2$ 中每个候选项集的支持度计数，如图 8.4 中的第 4 列。

(5) 确定频繁 2-项集的集合 $L_2$，它由具有最小支持度的 $C_2$ 中的候选 2-项集组成，见图 8.4 的第 5 列。

(6) 候选 3-项集的集合 $C_3$ 的产生，仍按(3)进行。得到候选集：

$$C_3 = \{\{A,B,C\}, \{A,B,E\}, \{A,C,E\}, \{B,C,D\}, \{B,C,E\}, \{B,D,E\}\}$$

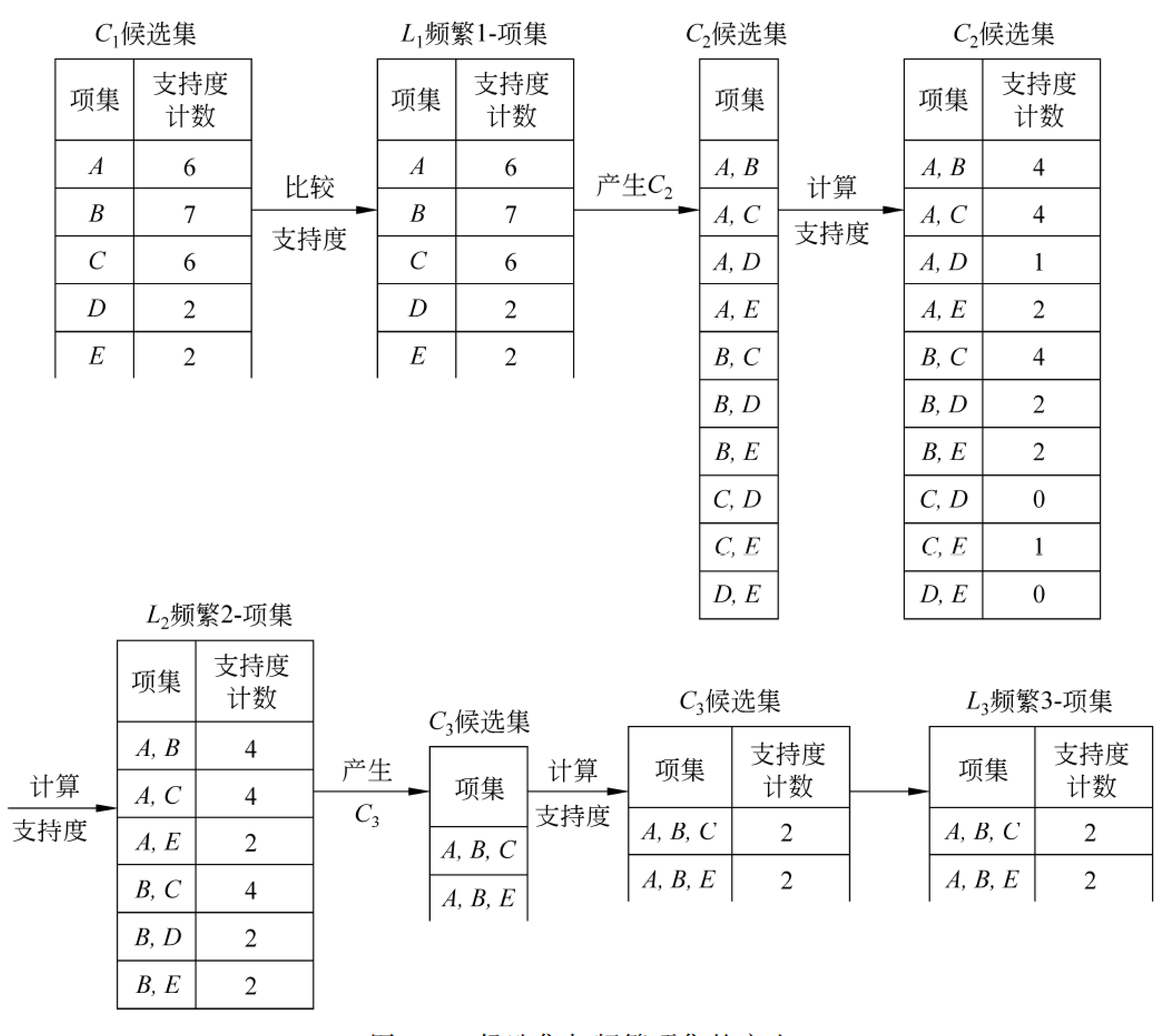
按 Apriori 性质，频繁项集的所有子集必须是频繁的。由于 $\{A,D\}$，$\{C,D\}$，$\{C,E\}$，$\{D,E\}$ 不是频繁项集，故 $C_3$ 中后 4 个候选不可能是频繁的，在 $C_3$ 中删除它们，见图 8.4 中第 6 列。

扫描 $D$ 中的事务，对 $C_3$ 中的候选项集计算支持度计数，见图 8.4 第 7 列。

(7) 确定 $L_3$，它由具有最小支持度的 $C_3$ 中候选 3-项集组成，见图 8.4 中的第 8 列。

(8) 按公式产生候选 4-项集的集合 $C_4$，产生结果$\{A,B,C,E\}$，这个项集被剪去，因为它的子集$\{B,C,E\}$不是频繁的。这样 $L_4=\varnothing$，此算法终止。$L_3$ 是最大的频繁项集，即$\{A,B,C\}$和$\{A,B,E\}$。

具体产生过程用图表示如图 8.4 所示。



$C_1$候选集

| 项集 | 支持度计数 |
|---|---|
| A | 6 |
| B | 7 |
| C | 6 |
| D | 2 |
| E | 2 |

比较支持度 →

$L_1$频繁1-项集

| 项集 | 支持度计数 |
|---|---|
| A | 6 |
| B | 7 |
| C | 6 |
| D | 2 |
| E | 2 |

产生$C_2$ →

$C_2$候选集

| 项集 |
|---|
| A, B |
| A, C |
| A, D |
| A, E |
| B, C |
| B, D |
| B, E |
| C, D |
| C, E |
| D, E |

计算支持度 →

$C_2$候选集

| 项集 | 支持度计数 |
|---|---|
| A, B | 4 |
| A, C | 4 |
| A, D | 1 |
| A, E | 2 |
| B, C | 4 |
| B, D | 2 |
| B, E | 2 |
| C, D | 0 |
| C, E | 1 |
| D, E | 0 |

计算支持度 →

$L_2$频繁2-项集

| 项集 | 支持度计数 |
|---|---|
| A, B | 4 |
| A, C | 4 |
| A, E | 2 |
| B, C | 4 |
| B, D | 2 |
| B, E | 2 |

产生$C_3$ →

$C_3$候选集

| 项集 |
|---|
| A, B, C |
| A, B, E |

计算支持度 →

$C_3$候选集

| 项集 | 支持度计数 |
|---|---|
| A, B, C | 2 |
| A, B, E | 2 |

→

$L_3$频繁3-项集

| 项集 | 支持度计数 |
|---|---|
| A, B, C | 2 |
| A, B, E | 2 |

图 8.4　候选集与频繁项集的产生

#### 4. 产生关联规则

由频繁项集产生关联规则的工作相对简单一点。根据前面提到的可信度的定义，关联规则的产生如下：

(1) 对于每个频繁项集 $L$，产生 $L$ 的所有非空子集；

(2) 对于 $L$ 的每个非空子集 $S$，如果$\frac{|L|}{|S|}\geqslant \text{min_conf}$，则输出规则“$S\rightarrow L-S$”。

说明：$L$-$S$ 表示在项集 $L$ 中除去 $S$ 子集的项集。$|L|$ 和 $|S|$ 表示项集 $L$ 和 $S$ 的在事务项目集中的计数。

由于规则由频繁项目集产生，因此每个规则都自动满足最小支持度。

在表 8.16 事务数据库中，频繁项集 $L=\{A,B,E\}$，可以由 $L$ 产生哪些关联规则？
$L$ 的非空子集 $S$ 有 $\{A, B\}$，$\{A, E\}$，$\{B, E\}$，$\{A\}$，$\{B\}$，$\{E\}$。可得到关联规则如下：

$$A \wedge B \rightarrow E \quad cf=2/4=50\%$$
$$A \wedge E \rightarrow B \quad cf=2/2=100\%$$
$$B \wedge E \rightarrow A \quad cf=2/2=100\%$$
$$A \rightarrow B \wedge E \quad cf=2/6=33\%$$
$$B \rightarrow A \wedge E \quad cf=2/7=29\%$$
$$E \rightarrow A \wedge B \quad cf=2/2=100\%$$

假设最小可信度为 60%，则最终输出的关联规则为

$$A \wedge E \rightarrow B \quad 100\%$$
$$B \wedge E \rightarrow A \quad 100\%$$
$$E \rightarrow A \wedge B \quad 100\%$$

对于频繁项集 $\{A,B,C\}$，同样可得其他关联规则。

### 8.3.3 Apriori 算法程序

为了生成所有频繁集，使用了递推的方法。程序包括 apriori-gen 子程序产生候选，完成连接和剪枝。has_infrequent_subset 子程序完成非频繁子集的测试。生成所有频繁项集的 Apriori 算法程序如下：

```
L1={1-itemsets};
for (K=2; LK-1≠Φ; K++)do
begin
    CK=apriori_gen(LK-1,min_sup);        //新的候选集
    for all transactions t∈D do
    begin
        Ct=subset(CK,t);                 //事务 t 中包含的候选集
        for all candidates c∈Ct   do
            c.count++;
    end
    LK={c∈CK|c.count>=min_sup}
end
Answer=∪LK;

Procedure apriori_gen( LK-1,min_sup)
CK=Φ
for each itemset li∈LK-1
    for each itemset lj∈LK-1
            if(li[1]=lj[1]) ∧ (li[2]=lj[2]) ∧ … ∧ (li[K-2]=lj[K-2]) ∧
            (li[K-1]<lj[K-1])
        then
            begin
            c=li join lj
```

```
            if has_infrequent_subset(c, L_{K-1})
                delete c;
            else add c to C_K;
        end
return C_K;

    Procedure has_infrequent_subset(c, L_{K-1})
for each (K-1)-subset s of c
    if s∉L_{K-1} then
        return TRUE;
return FALSE;
```

首先产生频繁 1-项集 $L_1$,然后是频繁 2-项集 $L_2$,直到有某个 $r$ 值使得 $L_r$ 为空,算法停止。这里在第 $K$ 次循环中,过程先产生候选 K-项集的集合 $C_K$,$C_K$ 中的每一个项集是对两个只有一个项不同的属于 $L_{K-1}$ 的频繁集做一个连接来产生的。$C_K$ 中的项集是用来产生频繁集的候选集,最后的频繁集 $L_K$ 必须是 $C_K$ 的一个子集。$C_K$ 中的每个元素需在交易数据库中进行验证来决定其是否加入 $L_K$,这里的验证过程是算法性能的一个瓶颈。这个方法要求多次扫描可能很大的交易数据库,即如果频繁集最多包含 10 个项,那么就需要扫描交易数据库 10 遍,这需要很大的 I/O 负载。

Agrawal 等引入了修剪技术来减小候选集 $C_K$ 的大小,由此可以显著地改进生成所有频繁集算法的性能。算法中引入的修剪策略基于 Apriori 性质:一个项集是频繁集当且仅当它的所有子集都是频繁集。那么,如果 $C_K$ 中某个候选项集有一个(K-1)-子集不属于 $L_{K-1}$,则这个项集可以被修剪掉不再被考虑,这个修剪过程可以降低计算所有的候选集的支持度的代价。J. Kleinberg 在文中,还引入 Hash 树(Hash Tree)方法来有效地计算每个项集的支持度。

### 8.3.4 基于 FP-tree 的关联规则挖掘算法

Apriori 算法存在一些固有的缺陷:

- 可能会产生大量的候选集。当长度为 1 的频繁集有 10 000 个的时候,长度为 2 的候选集个数将会超过 10M。还有就是如果要生成一个很长的规则的时候,要产生的中间元素也是巨大的。
- 必须多次重复扫描数据库,对候选集进行模式匹配,因此效率低下。

Jiawei Han 等人提出了一种基于 FP-树的关联规则挖掘算法 FP_growth,它采取"分而治之"的策略,将提供频繁项目集的数据库压缩成一棵频繁模式树(FP-树),但是仍然保留了项集关联信息,然后,将这种压缩后的数据库分成一组条件数据库,并分别挖掘每个数据库。理论和实验表明该算法优于 Apriori 算法。

#### 1. 算法描述

算法 FP_growth 将发现所有的频繁项目集的过程分为以下两步:构造频繁模式树 FP-树;调用 FP_growth 挖掘出所有的频繁项目集。在 FP-树中,每个结点由三个域组成:项目

名称 item_name、结点计数 count 和结点链(指针)。另外,为了方便树的遍历,利用频繁项集 $L_1$(1-项集),并增加"结点链",通过结点链指向该项目在树中的出现,即结点链头 head,指向 FP-树中与之名称相同的第一个结点。

下面仍利用上例事务数据库来说明 FP-树的构造过程和频繁模式挖掘过程。

(1) FP-树构造过程

数据库的第一次扫描与 Apriori 相同,它导出频繁项(1-项集)的集合,并得到它们的支持度计数。设最小支持度为 2,频繁项的集合按支持度计数的递减顺序排序,结果表记为 $L$。这样就有:

$$L = \{B:7, A:6, C:6, D:2, E:2\}$$

FP-树构造如下:首先,创建树的根结点,用 null 标记。第二次扫描事务数据库。每个事务中的项按 $L$ 中的次序处理(即按递减支持度计数排序)并对每个事务创建一个分支。

例如,第一个事务"$T_1$: $A,B,E$",按 $L$ 的次序包括三个项$\{B,A,E\}$,导致构造树的第一个分支 $<B:1, A:1, E:1>$。该分支具有三个结点,其中 $B$ 作根结点的子链接,$A$ 链接到 $B$,$E$ 链接到 $A$。从 $L$ 表中结点链中,项 $B$、$A$、$E$ 的指针分别指向树中 $B$、$A$、$E$ 结点。

第二个事务"$T_2$: $B,D$"按 $L$ 的次序也是$\{B,D\}$仍以 $B$ 开头,这样在 $B$ 结点中产生一个分支,该分支与 $T_1$ 项集存在路径共享前缀 $B$。这样,将结点 $B$ 的计数增加 1,即($B$: 2),并创造一个 $D$ 的新结点($D$: 1),作为($B$: 2)的子链接。

第三个事务"$T_3$: $B,C$"同第二个事务一样处理,因为有相同的 $B$ 为头,在 $B$ 结点又产生一个分支,产生新结点,记为($C$: 1),结点 $B$ 的计数再增加 1(为 3),即($B$: 3)。

第四个事务"$T_4$: $A,B,D$",按 $L$ 的次序为$\{B,A,D\}$。在 FP-树中 $B$,$A$,已有结点,将共享前缀路径,从 $A$ 结点分支产生 $D$ 的另一新结点,记为($D$: 1),共享结点 $B$,$A$ 的计数均增加 1,即($B$: 4),($A$: 2)。此($D$: 1)结点用指针指向前面产生的($D$: 1)结点,在 $L$ 表中结点链接中指针指向该($D$: 1)结点。

第五个事务"$T_5$: $A,C$",按 $L$ 表的次序为$\{A,C\}$。在 FP-树中,由于该事务不含 $B$ 结点,不能共享 $B$ 分支。从 null 结点产生 FP-树的第二个分支,建新 $A$ 结点,记为($A$: 1),由该结点产生分支,建新 $C$ 结点,即为($C$: 1)。由于 $B$ 分支中有($A$: 2)结点。这样,从($A$: 2)结点用指针指向此($A$: 1)结点,$B$ 分支中有($C$: 1)结点,它用指针指向此($C$: 1)结点。

第六个事务"$T_6$: $B,C$",同第三个事务那样,沿 FP-树的 $B$-$C$ 分支的结点计数各增加 1,变为($B$: 5)和($C$: 2)。

第七个事务"$T_7$: $A,C$",同第五个事务,沿 FP-树的 $A$-$C$ 分支的结点计数各增加 1,变为($A$: 2)和($C$: 2)。

第八个事务"$T_8$: $A,B,C,E$",按 $L$ 表的次序为$\{B,A,C,E\}$,可沿分支 $B$-$A$ 方向,在 $A$ 结点处新建分支,建 $C$ 结点,记($C$: 1),由该结点再建分支,建 $E$ 结点,记为($E$,1),前面 $B$,$A$ 结点计数各增加 1,变为:($B$: 6),($A$: 3)。FP-树中原 $E$ 结点($E$: 1)中的指针指向该($E$,1)结点。

第九个事务"$T_9$: $A,B,C$",按 $L$ 表的次序为$\{B, A, C\}$,同第八个事务,分支 $B$-$A$-$C$ 方向,且已有结点,分别对 $B$,$A$,$C$ 三个结点计数增加 1,变为($B$: 7),($A$: 4),($C$: 2)。最终的 FP-树的表示如图 8.5 所示。

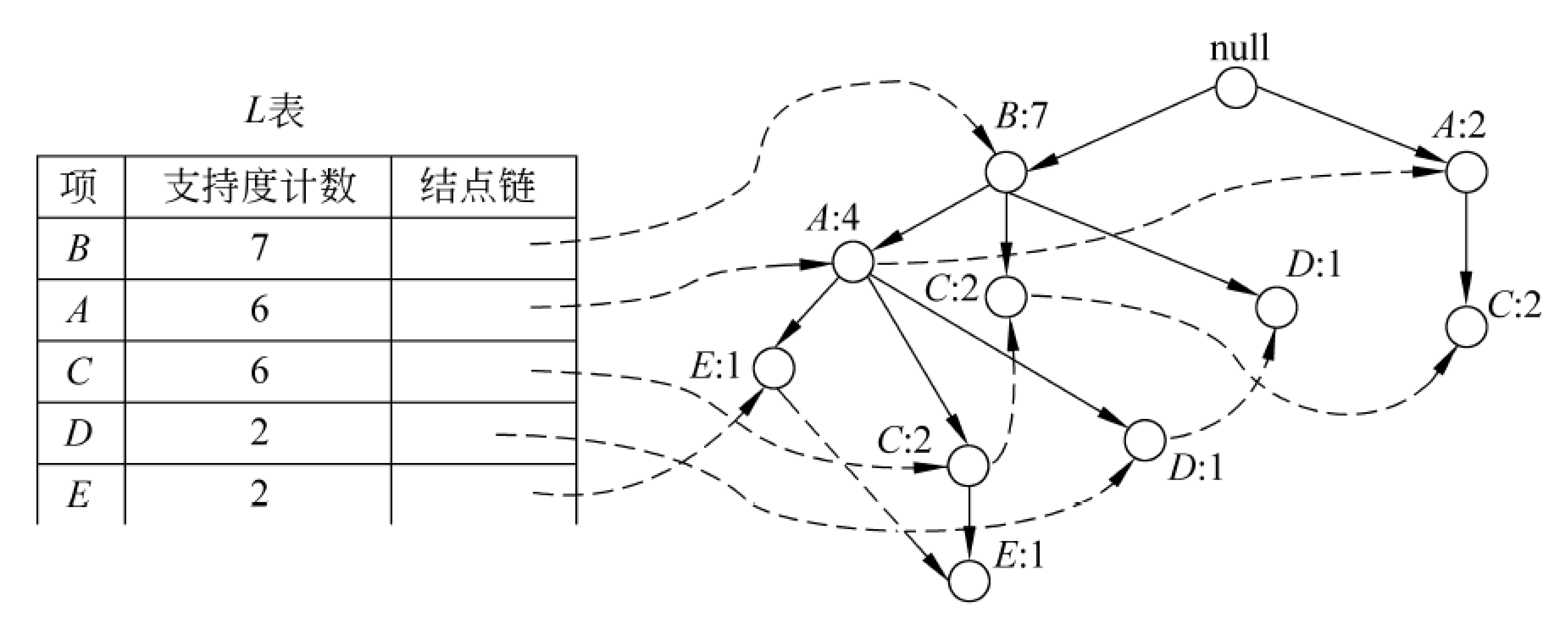


图 8.5　表 8.16 事务数据库的 FP-树

从 FP-树可以看出，从 L 表的结点连的指针开始，指向 B 结点，它的计数器为 7，指向 A 结点，共有两个 A 结点，累加计数为 6；指向 C 结点，共有三个 C 结点，累加计数为 6；指向 D 结点，共有二个 D 结点，累加计数为 2；指向 E 结点，共有二个 E 结点，累加计数为 2。这样，频繁模式都在 FP-树中表现了出来。

(2) 频繁模式挖掘过程

从 FP-树中来挖掘频繁模式，先从 L 表中最后一项开始。E 在 FP-树有两个分支，路径为<BAE：1>和<BACE：1>。以 E 为后缀，它的两个对应前缀路径是(BA：1)和(BAC：1)，它们形成 E 的条件模式基。它的条件 FP-树只包含单个路径<B：2，A：2>；不包含 C，因为它的支持度计数为 1，小于最小支持度计数。该单个路径产生频繁模式的所有组合：{BE：2，AE：2，BAE：2}。

对于 D，它的两个前缀形成条件模式基{(BA：1)，(B：1)}，产生一个单结点的条件 FP-树(B：2)，并导出一个频繁模式{BD：2}。

对于 C，它的条件模式基是{(BA：2)，(B：2)，(A：2)}，它的条件 FP-树有两个分支(B：4，A：2)和(A：2)。它的频繁模式集为：{BC：4，AC：4，BAC：2}。

对于 A，它的条件模式基是{(B：4)}，它的 FP-树只包含一个结点(B：4)，产生一个频繁模式{BA：4}，如表 8.7 所示。

**表 8.7　利用 FP-树挖掘频繁模式**

| 项 | 条件模式基 | 条件 FP-树 | 频繁模式 |
|---|---|---|---|
| E | BA：1，BAC：1 | (B：2，A：2) | BE：2，AE：2，BAE：2 |
| D | BA：1，B：1 | (B：2) | BD：2 |
| C | BA：2，B：2，A：2 | (B：4，A：2)(A：2) | BC：4，AC：4，BAC：2 |
| A | B：4 | (B：4) | BA：4 |

### 2. 基于 FP-树算法

(1) 构造频繁模式树算法

① 扫描事务数据库 D 一次。收集频繁项的集合(1-项集)以及相应的支持度。按照支

持度降序排序，构成频繁项表 $L$。

② 创建 FP-树的根结点，以 null 标记。对于 $D$ 中的每个事务 $T$，进行如下处理：选择 $T$ 中的频繁项目，并按照 $L$ 中的次序排列。设排列之后的频繁项表为 $[p|P]$，其中 $p$ 是第一个项目，$P$ 是剩余的项目表；如果 $[p|P]$ 非空，调用 insert_tree($[p|P]$, $T$)。

insert_tree($[p|P]$, $T$)的执行过程如下：

如果 $T$ 有子女 $N$ 使得 N. item_name=p. item_name，则 $N$ 的计数加 1；否则创建一个新结点 $N$，将其计数设置为 1，链接到它的父结点 $T$，并且通过结点链将其链接到具有相同 item_name 的结点。如果 $P$ 非空，则递归地调用 insert_tree($P$, $T$)。

(2) 挖掘频繁项目集算法

FP-树的频繁项目集挖掘通过调用 FP_growth(FP-tree, null)实现。该实现过程如下：

Procedure FP_growth(Tree, $\alpha$)

① 如果 Tree 含单个路径 $P$，则

② 　　对于路径 $P$ 中的每个组合(记作 $\beta$)

③ 　　产生模式 $\beta \cup \alpha$，其支持度 support=$\beta$ 中结点的最小支持度。

④ 否则 对于在 Tree 头部的每个 $a_i$

⑤ 　　产生一个模式 $\beta = a_i \cup \alpha$，其支持度 support=$a_i$ 的支持度

⑥ 　　构造 $\beta$ 的条件模式基，然后构造 $\beta$ 的条件 $Tree_\beta$

⑦ 　　如果 $Tree_\beta$ 非空，则调用 FP_growth($Tree_\beta$, $\beta$)

FP_growth 方法将发现长频繁模式的问题转换为递归地发现一些短模式，然后连接后缀。它使用最不频繁的项做后缀，提供了非常好的选择性，大大降低了搜索开销。

对 FP_growth 算法的性能研究表明：对于挖掘长的和短的频繁模式，它都是有效的和可伸缩的，并且大约比 Apriori 算法快一个数量级。

### 3. 示例说明

例如，假设有 10 个事务的数据库 $D$，项目集合 $\{a, b, c, d, e, f, g, h, i\}$，最小支持度 20%，如表 8.8 所示。

**表 8.8　事务数据库**

| TID | $T_0$ | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ | $T_7$ | $T_8$ | $T_9$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 项集 | $e$ | $a,c,g,i$ | $d,h$ | $b,d$ | $d,e$ | $a,c,e,i$ | $a,c,e,f,i$ | $a,e,g$ | $a,c,e,i$ | $c,e,g$ |

数据库 $D$ 对应的频繁模式树 FP-树如图 8.6 所示。

使用 FP_growth 算法，可以得到数据库 $D$ 的频繁项目集为 $\{\{e\}: 7, \{a\}: 5, \{c\}: 5, \{i\}: 4, \{d\}: 3, \{g\}: 3, \{a,c\}: 4, \{a,e\}: 4, \{a,g\}: 2, \{a,i\}: 4, \{c,e\}: 4, \{c,g\}: 2, \{c,i\}: 4, \{e,g\}: 2, \{e,i\}: 3, \{a,c,e\}: 3, \{a,c,i\}: 4, \{a,e,i\}: 3, \{c,e,i\}: 3, \{a,c,e,i\}: 3\}$。其中，$b$，$f$，$h$ 不是频繁项集。

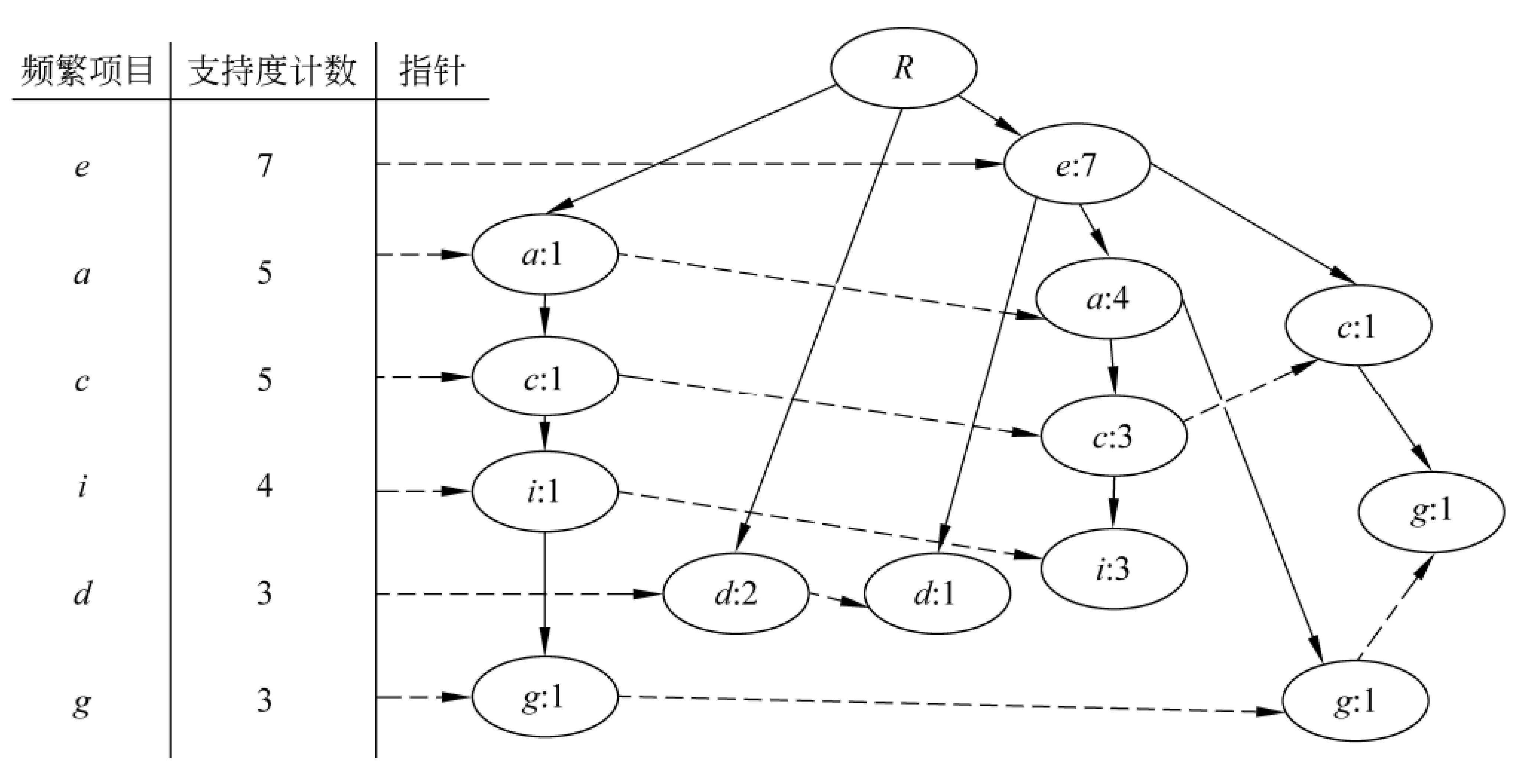



图 8.6　FP-树实例

# 习　题　8

1. 说明等价关系、等价类以及划分的定义。

2. 说明集合 $X$ 的上、下近似关系定义。

3. 说明正域、负域和边界的定义。

4. 说明粗糙集定义和确定度定义。

5. 什么是属性约简？

6. 什么是属性集的核？

7. 请用粗糙集的条件属性相对于决策属性的约简定义，对于两类人数据库表 6.3 进行属性约简计算。

8. 说明条件属性 $C$ 与决策属性 $D$ 之间的依赖度 $\gamma(C,D)$ 的含义是什么？

9. 依赖度 $\gamma(C,D)$ 的性质是什么？

10. 属性 $a$ 的重要度 $\mathrm{SGF}(a,C,D)$ 的含义是什么？

11. 最小属性集的概念是什么？

12. 在数据库中获得最小属性集的步骤是什么？

13. 如何利用集合之间的上下近似关系获得规则？

14. 按照聚类的原理和方法划分有哪三种聚类算法？各种聚类算法的思想是什么？

15. 写出 K-均值聚类算法的计算步骤。

16. 规则的支持度和可信度的含义是什么？

17. 关联规则的兴趣度定义是什么？说明兴趣度的作用。

18. 数据库有 4 个事务。设最小支持度为 50%。

| TID | 项 | TID | 项 |
| --- | --- | --- | --- |
| $T_1$ | $A$, $C$, $D$ | $T_3$ | $A$, $B$, $C$, $E$ |
| $T_2$ | $B$, $C$, $E$ | $T_4$ | $B$, $E$ |

使用 Apriori 算法找出所有的频繁项目集。

19. 实现 Apriori 算法,说明 Apriori 算法的主要系统开销在哪里?

20. 对上述事务集使用 FP-树算法找出所有的频繁项目集,并比较二者在性能上的差异。

21. 对表 8.8 事务数据库,利用 FP-树算法进行详细计算,得出图 8.6 所示的 FP-树。

22. 对上题得出的频繁项集,求出关联规则。

23. 集合论原理用于分类问题的思想是什么?

24. 集合论原理用于聚类问题的思想是什么?

25. 集合论原理用于关联规则挖掘的思想是什么?

# 第9章 神经网络

## 9.1 神经网络概念与感知机

### 9.1.1 神经网络原理

#### 1. 人工神经网络概念

神经生理学家和神经解剖学家早已证明，人的思维是通过人脑完成的，神经元是组成人脑的最基本单元，人脑神经元大约有 $10^{11}$～$10^{12}$个（约 1000～10 000 亿个）。

神经元由细胞体、树突和轴突三部分组成，是一种根须状的蔓延物。神经元的中心有一闭点，称为细胞体，它能对接收到的信息进行处理。细胞体周围的纤维有两类，轴突是较长的神经纤维，是发出信息的。树突的神经纤维较短，而分支很多，是用于接收信息的。一个神经元的轴突末端与另一个神经元的树突之间密切接触，传递神经元冲动的地方称为突触。经过突触的冲动传递是有方向性的，不同的突触进行的冲动传递效果不一样，有的使后一神经元发生兴奋，有的使它受到抑制。每个神经元可有 10～$10^4$ 个突触。这表明大脑是一个广泛连接的复杂网络系统。从信息处理功能看，神经元具有如下性质：

(1) 多输入单输出；

(2) 突触兼有兴奋和抑制两种性能；

(3) 可时间加权和空间加权；

(4) 可产生脉冲；

(5) 脉冲进行传递；

(6) 非线性(有阈值)。

神经元的数学模型用图 9.1 表示。

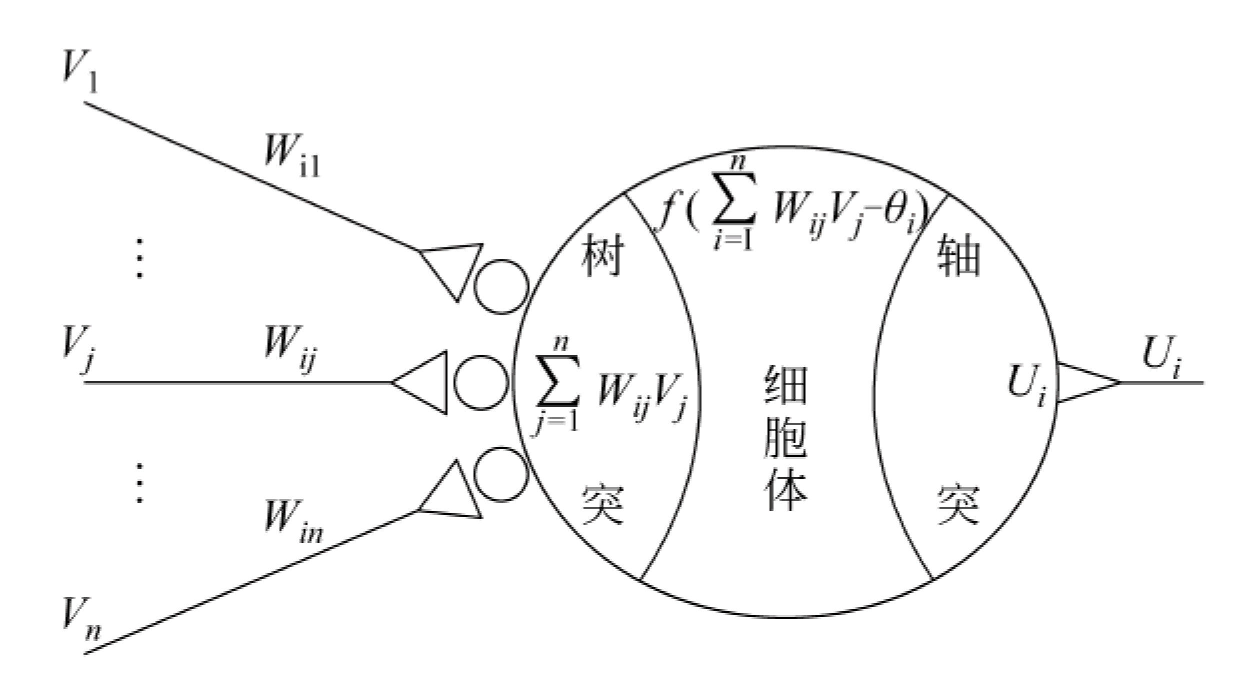



图 9.1 神经元模型

其中 $V_1, V_2, \cdots, V_n$ 为输入；$U_i$ 为该神经元的输出；$W_{ij}$ 为外面神经元与该神经元连接强度(即权)，$\theta$ 为阈值，$f(X)$ 为该神经元的作用函数。

**2. MP 模型与 Hebb 规则**

(1) MP(Mcculloch,Pitts)模型

每个神经元的状态 $U_i(i=1,2,\cdots,n)$只取 0 或 1,分别代表抑制与兴奋。每个神经元的状态,由 M-P 方程决定:

$$U_i = f\left(\sum_j w_{ij} V_j - \theta_j\right) \quad i = 1,2,\cdots,n \tag{9.1}$$

其中 $W_{ij}$ 是神经元之间的连接强度,$W_{ii}=0$,$W_{ij}(i\neq j)$是可调实数,由学习过程来调整。$\theta_i$ 是阈值,$f(x)$是阶梯函数。

MP 模型实质上是把人脑神经元的功能,转换成了数学模型。以后就用这个数学模型去解决非生物中模式识别的分类问题。

(2) Hebb 规则

Hebb 学习规则:若 $i$ 与 $j$ 两种神经元之间同时处于兴奋状态,则它们间的连接应加强,即

$$\Delta W_{ij} = \alpha U_i V_j \quad (\alpha > 0) \tag{9.2}$$

设 $\alpha=1$,当 $U_i=V_j=1$ 时,$\Delta W_{ij}=1$,在 $U_i$,$V_j$ 中有一个为 0 时,$\Delta W_{ij}=0$。这一规则与“条件反射”学说一致,并得到神经细胞学说的证实。

**3. 各种作用函数**

(1) [0,1]阶梯函数

$$f(x) = \begin{cases} 1, & x > 0 \\ 0, & x \leqslant 0 \end{cases} \tag{9.3}$$

(2) [-1,1]的阶梯函数

$$f(x) = \begin{cases} 1, & x > 0 \\ -1, & x \leqslant 0 \end{cases} \tag{9.4}$$

(3) (-1,1)S 型函数

$$f(x) = \frac{1-\mathrm{e}^{-x}}{1+\mathrm{e}^{-x}} \tag{9.5}$$

(4) (0,1)S 型函数

$$f(x) = \frac{1}{1+\mathrm{e}^{-x}} \tag{9.6}$$

(0,1)S 型函数如图 9.2 所示。

## 9.1.2 感知机网络

感知机网络是神经网络应用最早,且最成功的神经网络模型。

**1. 感知机(Perceptron)原理**

感知机网络是双层模型,其结构如图 9.3 所示。

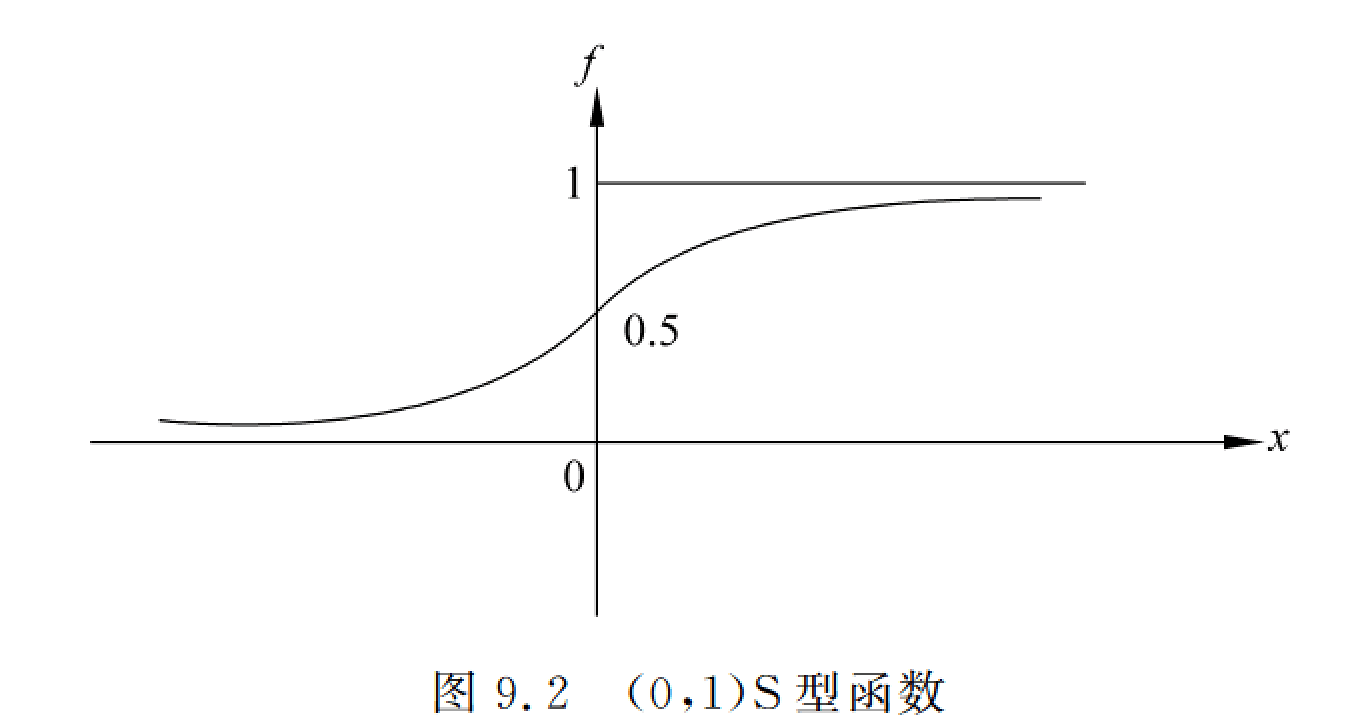


图 9.2 (0,1)S 型函数

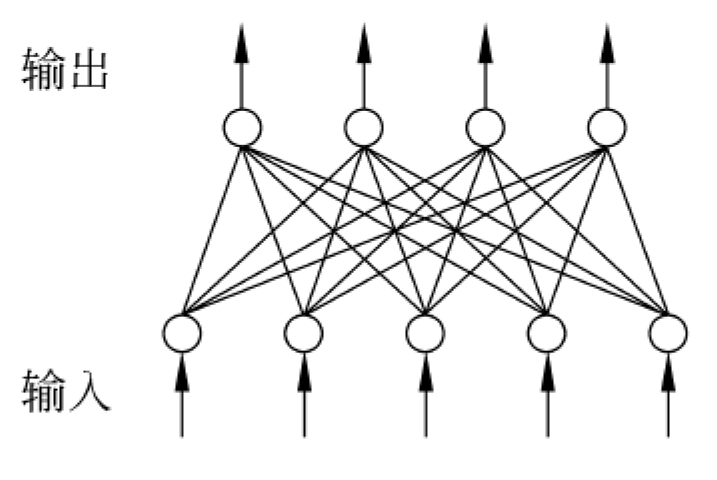


图 9.3 感知机网络结构

输出层神经元 $i$ 的输入为

$$I_i = \sum W_{ij} x_j - \theta_i \tag{9.7}$$

$x_j$ 为输入层 $j$ 神经元的输出，$W_{ij}$ 为输入层神经元 $j$ 到输出层神经元 $i$ 的连接权值。输出层神经元 $i$ 的输出为

$$O_i = f(I_i) \tag{9.8}$$

其中 $f(x)$ 为神经元作用函数，感知机采用[0,1]阶梯函数。

设 $i$ 神经元的实际输出为 $D_i$，它与计算输出 $O_i$ 之差为

$$\delta_i = D_i - O_i \tag{9.9}$$

通过样本学习，修正权值 $W_{ij}$ 使 $\delta_i$ 尽可能小。利用著名的德尔塔规则(delta rule)计算：

$$\Delta W_{ij} = \alpha \delta_i x_j \quad (\alpha \text{ 为常数}) \tag{9.10}$$

$$W_{ij}(t+1) = W_{ij}(t) + \Delta W_{ij} \tag{9.11}$$

阈值修正公式

$$\Delta\theta_i = \alpha\delta_i \tag{9.12}$$

$$\Delta\theta_i(t+1) = \theta_i(t) + \Delta\theta_i \tag{9.13}$$

更新权值 $W_{ij}$ 和 $\theta_i$。对样本重复以上计算，经过多次反复修正，将使 $\delta_i$ 趋向于 0。

**2. 感知机网络的实现**

(1) 数据结构

① 输入结点向量 $\boldsymbol{X}$(结点数为 $m$)

$$x_1, x_2, \cdots, x_m$$

② 输出结点向量(结点数为 $n$)

| 结点 | 1 | 2 | 3 | … | $n$ |
|---|---|---|---|---|---|
| 输入 | $I_1$ | $I_2$ | $I_3$ | … | $I_n$ |
| 计算输出 | $O_1$ | $O_2$ | $O_3$ | … | $O_n$ |
| 实际输出 | $D_1$ | $D_2$ | $D_3$ | … | $D_n$ |

③ 网络上权值

$$W = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nm} \end{bmatrix}$$

(2) 学习过程

给出一组学习样本(共 $P$ 个)

$$(X(1),D(1)),(X(2),D(2)),\cdots,(X(p),D(p))$$

对第 $k$ 个样本$(X(k),D(k))$有:

输入:

$$X(k)=(x_1(k),x_2(k),\cdots,x_m(k))$$

实际输出:

$$D(k)=(D_1(k),D_2(k),\cdots,D_n(k))$$

① 给网络上权值和阈值赋初值,如:

$$W_{ij}\equiv 0,\quad \theta_i\equiv 0$$

样本循环变量赋初值:$k=1$,总误差初值 $E=0$,迭代次数 $L=0$。

② 通过感知机模型公式计算,对第 $k$ 个样本:

输入:

$$X(k)=(x_1(k),x_2(k),\cdots,x_m(k))$$

计算输出:

$$O(k)=(O_1(k),O_2(k),\cdots,O_n(k))$$

迭代次数 $L=L+1$

③ 误差计算

每个输出结点误差:

$$\delta_i(k)=D_i(k)-O_i(k)\quad (i=1,2,\cdots,n)$$

第 $k$ 个样本误差:

$$e_k=\sum_{i=1}^{n}|\delta_i(k)|$$

④ 权值修正

原则:修正权 $W_{ij}$ 使 $\delta_i$ 尽可能的小,利用德尔塔规则(Delta Rule),即:

$$\Delta W_{ij}=\alpha\delta_i(k)\,x_j(k)$$

$$W_{ij}(n+1)=W_{ij}(n)+\Delta\omega_{ij}$$

⑤ 阈值修正

$$\Delta\theta_i=\alpha\delta_i$$

$$\Delta\theta_i(t+1)=\theta_i(t)+\Delta\theta_i$$

⑥ 计算 $P$ 个样本的总误差

$$E=E+e_k$$

⑦ 检查

$$k=P?$$

是:检查$|E|\leqslant 0.05$?

是:计算结束。输出迭代次数 $L$ 和总误差 $E$,输出网络权值 $W_{ij}$。

否:$k=1,E=0$。样本再次学习,转②循环。

否:$k=k+1$,做下一个样本,转②循环。

### 9.1.3 感知机实例与讨论

#### 1. 感知机实例

两值逻辑加法例,输入数据和输出数据样本如下:

| 输入 $x1$ | $x2$ | 输出 $d$ (实际) |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

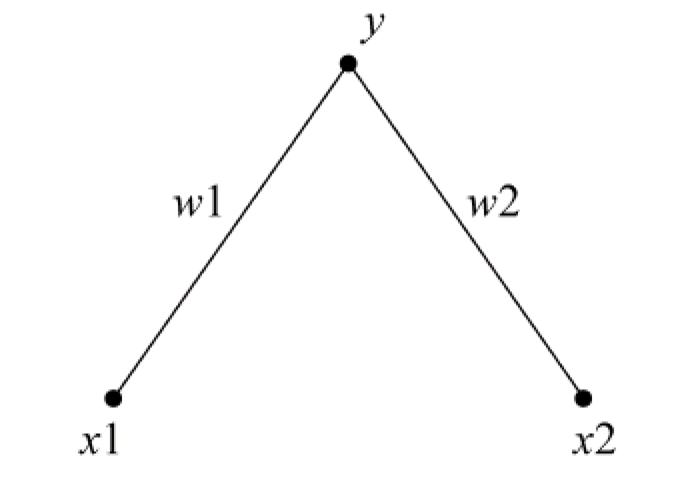


图 9.4 两值逻辑加法神经网络

该例的神经网络结构如图 9.4 所示。

该例子的感知机计算公式:

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(k+1)} = \begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(k)} + c(d-y)\begin{bmatrix} x1 \\ x2 \end{bmatrix}$$

初值:

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad c = 1$$

其中 $d$ 为期望输出,$y$ 为计算输出。

计算过程:

$$K = 1, \quad y = f(0+0) = 0$$

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(1)} = \begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(0)} + (0-0)\cdot\begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$K = 2, \quad y = f(0+0) = 0$$

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(2)} = \begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(1)} + (1-0)\cdot\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$K = 3, \quad y = f(0+0) = 0$$

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(3)} = \begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(2)} + (1-0)\cdot\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$K = 4, \quad y = f(1+1) = f(2) = 1$$

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(4)} = \begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(3)} + (1-1)\cdot\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

再循环一次,将会得到所有例子的$(d-y)$值均为零,即权值$(w1=1, w2=1)$满足所有实例要求。

#### 2. 感知机讨论

现将二值逻辑加法实例,改为异或问题实例,即第四个样本的实际输出值由 1 改为 0。异或问题样本示意图如图 9.5 所示。

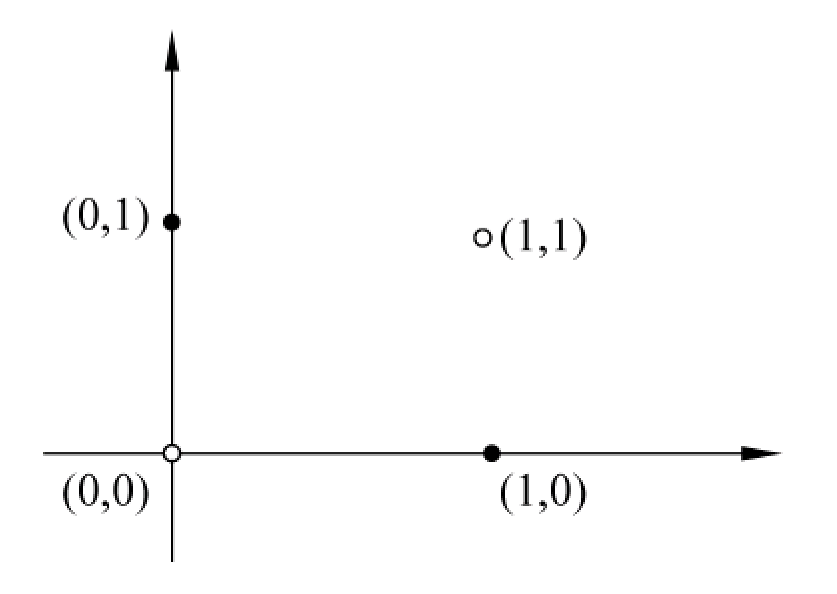
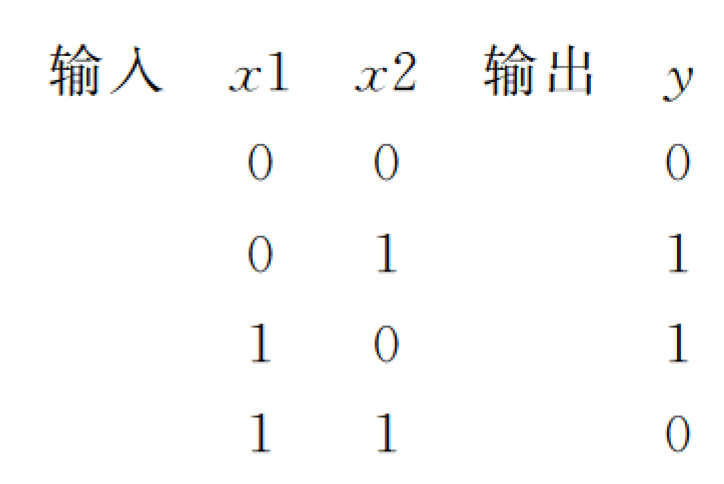



图 9.5　异或问题样本示意图



| 输入 | $x1$ | $x2$ | 输出 | $y$ |
|---|---|---|---|---|
| | 0 | 0 | | 0 |
| | 0 | 1 | | 1 |
| | 1 | 0 | | 1 |
| | 1 | 1 | | 0 |

感知机对异或问题的神经网络计算如下：

$K=1,2,3$ 的计算同二值逻辑加法样本计算。

$K=4$ 时有：

$$y=f(1+1)=f(2)=1$$

$$\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(4)}=\begin{bmatrix} w1 \\ w2 \end{bmatrix}^{(3)}+(0-1)\cdot\begin{bmatrix} 1 \\ 1 \end{bmatrix}=\begin{bmatrix} 1 \\ 1 \end{bmatrix}-\begin{bmatrix} 1 \\ 1 \end{bmatrix}=\begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

修改后的权值，又回到了初始状态，如果继续计算，将出现无限循环，永远都不会收敛。

该例充分说明感知机对异或问题(非线性)无效。要解决非线性问题，需要在输入、输出两层神经网络中间增加隐结点层。下面讨论的反向传播模型(BP)可以解决非线性问题。

## 9.2　反向传播网络

### 9.2.1　反向传播网络结构

反向传播(Back Propagation，BP)网络是 1985 年由 Rumelhart 等人提出的。

**1. 多层网络结构**

神经网络不仅有输入结点、输出结点，而且有一层或多层隐结点，如图 9.6 所示。

**2. 作用函数为(0,1)S 型函数**

$$f(x)=\frac{1}{1+\mathrm{e}^{-x}} \tag{9.14}$$

输出层

隐层

输入层

图 9.6　BP 模型网络结构

**3. 误差函数**

对第 $p$ 个样本误差计算公式为

$$E_p=\frac{1}{2}\sum_i (t_{pi}-O_{pi})^2 \tag{9.15}$$

其中 $t_{pi}$、$O_{pi}$ 分别是样本实际输出与计算输出。

### 9.2.2　BP 网络学习公式推导

BP 网络表示为，输入结点：$x_j$，隐结点：$y_i$，输出结点 $O_l$。

输入结点与隐结点间的网络权值为 $W_{ij}$，隐结点与输出结点间的网络权值为 $T_{li}$。当输出结点的实际输出为 $t_l$ 时，BP 模型的计算公式为

**1. 隐结点的输出**

$$y_i = f\left(\sum_j w_{ij}x_j - \theta_i\right) = f(\text{net}_i)$$

其中，$\text{net}_i = \sum_j w_{ij}x_j - \theta_i$。

**2. 输出结点计算输出**

$$O_l = f\left(\sum_i T_{li}y_i - \theta_l\right) = f(\text{net}_l)$$

其中，$\text{net}_l = \sum_i T_{li}y_i - \theta_l$。

**3. 输出结点的误差公式**

$$E = \frac{1}{2}\sum_l (t_l - O_l)^2 = \frac{1}{2}\sum_l \left(t_l - f\left(\sum_i T_{li}y_i - \theta_l\right)\right)^2$$

$$= \frac{1}{2}\sum_l (t_l - f(\sum_i T_{li} f(\sum_j w_{ij}x_j - \theta_i) - \theta_l))^2$$

**4. 对网络权值修正公式的推导**

(1) 对输出结点的公式推导

$$\frac{\partial E}{\partial T_{li}} = \sum_{k=1}^{n} \frac{\partial E}{\partial O_k}\frac{\partial O_k}{\partial T_{li}} = \frac{\partial E}{\partial O_l}\frac{\partial O_l}{\partial T_{li}}$$

$E$ 是多个 $O_k$ 的函数，但只有一个 $O_l$ 与 $T_{li}$ 有关，各 $O_k$ 间相互独立。其中

$$\frac{\partial E}{\partial O_l} = \frac{1}{2}\sum_k -2(t_k - O_k)\cdot\frac{\partial O_k}{\partial O_l} = -(t_l - O_l)$$

$$\frac{\partial O_l}{\partial T_{li}} = \frac{\partial O_l}{\partial \text{net}_l}\cdot\frac{\partial \text{net}_l}{\partial T_{li}} = f'(\text{net}_l)\cdot y_i$$

则

$$\frac{\partial E}{\partial T_{li}} = -(t_l - O_l)\cdot f'(\text{net}_l)\cdot y_i \tag{9.16}$$

设输出结点误差

$$\delta_l = (t_l - O_l)\cdot f'(\text{net}_l) \tag{9.17}$$

则

$$\frac{\partial E}{\partial T_{li}} = -\delta_l y_i \tag{9.18}$$

(2) 对隐结点的公式推导

$$\frac{\partial E}{\partial W_{ij}} = \sum_l\sum_i \frac{\partial E}{\partial O_l}\frac{\partial O_l}{\partial y_i}\frac{\partial y_i}{\partial W_{ij}}$$

$E$ 是多个 $O_l$ 函数，针对某一个 $W_{ij}$，对应一个 $y_i$，它与所有 $O_l$ 有关，其中：

$$\frac{\partial E}{\partial O_l} = \frac{1}{2}\sum_k -2(t_k - O_k)\cdot\frac{\partial O_k}{\partial O_l} = -(t_l - O_l)$$

$$\frac{\partial O_l}{\partial y_i} = \frac{\partial O_l}{\partial \text{net}_l} \cdot \frac{\partial \text{net}_l}{\partial y_i} = f'(\text{net}_l) \cdot \frac{\partial \text{net}_l}{\partial y_i} = f'(\text{net}_l) \cdot T_{li}$$

$$\frac{\partial y_i}{\partial W_{ij}} = \frac{\partial y}{\partial \text{net}_i} \cdot \frac{\partial \text{net}_i}{\partial W_{ij}} = f'(\text{net}_i) \cdot x_j$$

则

$$\begin{aligned}\frac{\partial E}{\partial W_{ij}} &= -\sum_l (t_l - O_l) f'(\text{net}_l) \cdot T_{li} \cdot f'(\text{net}_i) x_j \\ &= -\sum_l \delta_l T_{li} \cdot f'(\text{net}_i) \cdot x_j \end{aligned} \tag{9.19}$$

设隐结点误差

$$\delta'_i = f'(\text{net}_i) \cdot \sum_l \delta_l T_{li} \tag{9.20}$$

则

$$\frac{\partial E}{\partial W_{ij}} = -\delta'_i x_j \tag{9.21}$$

由于权值的修正 $\Delta T_{li}$，$\Delta W_{ij}$ 正比于误差函数沿梯度下降，则有：

$$\Delta T_{li} = -\eta \frac{\partial E}{\partial T_{li}} = \eta \delta_l y_i \tag{9.22}$$

$$\delta_l = (t_l - O_l) \cdot f'(\text{net}_l) \tag{9.23}$$

$$\Delta W_{ij} = -\eta' \frac{\partial E}{\partial W_{ij}} = \eta' \delta_i x_j \tag{9.24}$$

$$\delta_i = f'(\text{net}_i) \sum_l \delta_l T_{li} \tag{9.25}$$

(3) 公式推导结果汇总

① 对输出结点误差：

$$\delta_l = (t_l - O_l) \cdot f'(\text{net}_l) \tag{9.26}$$

② 输出层网络权值修正：

$$T_{li}(k+1) = T_{li}(k) + \Delta T_{li} = T_{li}(k) + \eta \delta_l y_i \tag{9.27}$$

③ 对隐结点误差：

$$\delta'_i = f'(\text{net}_i) \cdot \sum_l \delta_l T_{li} \tag{9.28}$$

④ 隐结点网络权值修正：

$$W_{ij}(k+1) = W_{ij}(k) + \Delta W_{ij} = W_{ij}(k) + \eta' \delta'_i x_j \tag{9.29}$$

其中，隐结点误差 $\delta'_i$ 的含义：

$\sum_l \delta_l T_{li}$ 表示输出层结点 $l$ 的误差 $\delta_l$ 通过权值 $T_{li}$ 向隐结点 $i$ 反向传播（误差 $\delta_l$ 乘权值 $T_{li}$ 再累加）成为隐结点 $i$ 的误差，如图 9.7 所示。

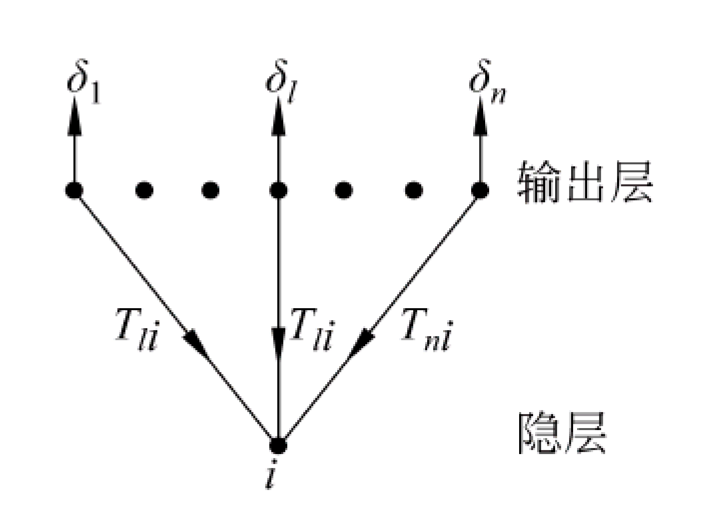



图 9.7 误差反向传播示意图

**5. 阈值的修正**

阈值 $\theta$ 也是一个变化值，在修正权值的同时也修正它，原理同权值的修正。

（1）对输出结点的公式推导

$$\frac{\partial E}{\partial \theta_l} = \frac{\partial E}{\partial O_l}\frac{\partial O_l}{\partial \theta_l}$$

其中$\frac{\partial E}{\partial O_l}=-(t_l-O_l)$，对某个 $\theta_l$ 对应一个 $O_l$

$$\frac{\partial O_l}{\partial \theta_l} = \frac{\partial O_l}{\partial \text{net}_l}\cdot\frac{\partial \text{net}_l}{\partial \theta_l} = f'(\text{net}_l)\cdot(-1)$$

则

$$\frac{\partial E}{\partial \theta_l} = (t_l - O_l)\cdot f'(\text{net}_l) = \delta_l \tag{9.30}$$

由于

$$\Delta\theta_l = \eta\frac{\partial E}{\partial \theta_l} = \eta\delta_l$$

则

$$\theta_l(k+1) = \theta_l(k) + \eta\delta_l \tag{9.31}$$

（2）对隐结点的公式推导

$$\frac{\partial E}{\partial \theta_i} = \frac{\partial E}{\partial y_i}\cdot\frac{\partial y_i}{\partial \theta_i} = \frac{\partial E}{\partial O_l}\frac{\partial O_l}{\partial y_i}\frac{\partial y_i}{\partial \theta_i}$$

其中：

$$\frac{\partial E}{\partial O_l} = -\sum_l(t_l - O_l)$$

$$\frac{\partial O_l}{\partial y_i} = f'(\text{net}_l)\cdot T_{li}$$

$$\frac{\partial y_i}{\partial \theta_i} = \frac{\partial y}{\partial \text{net}_l}\cdot\frac{\partial \text{net}_l}{\partial \theta_i} = f'(\text{net}_i)\cdot(-1) = -f'(\text{net}_i)$$

则

$$\frac{\partial E}{\partial \theta_i} = \sum_l(t_l - O_l)f'(\text{net}_l)\cdot T_{li}\cdot f'(\text{net}_i) = \sum_l\delta_l T_{li}\cdot f'(\text{net}_i) = \delta'_i \tag{9.32}$$

由于

$$\Delta\theta_i = \eta\frac{\partial E}{\partial \theta_i} = \eta'\delta'_i$$

则

$$\theta_i(k+1) = \theta_i(k) + \eta'\delta'_i \tag{9.33}$$

**6. 作用函数 $f(x)$ 的导数公式**

函数 $f(x)=\frac{1}{1+\mathrm{e}^{-x}}$，存在关系

$$f'(x)=f(x)\cdot(1-f(x))$$

则

$$f'(\text{net}_k) = f(\text{net}_k)\cdot(1 - f(\text{net}_k)) \tag{9.34}$$

对输出结点：

$$O_l = f(\text{net}_l)$$

$$f'(\text{net}_l) = O_l(1-O_l) \tag{9.35}$$

对隐结点：

$$y_i = f(\text{net}_i)$$
$$f'(\text{net}_i) = y_i(1-y_i) \tag{9.36}$$

**7. BP 模型计算公式汇总**

（1）输出结点的输出 $O_l$ 计算公式

① 输入结点的输入 $x_j$

② 隐结点的输出：

$$y_i = f\left(\sum_j W_{ij} X_j - \theta_i\right)$$

其中：连接权值 $W_{ij}$，结点阈值 $\theta_i$。

③ 输出结点的输出：

$$O_l = f\left(\sum_i T_{li} y_i - \theta_l\right)$$

其中：连接权值 $T_{li}$，结点阈值 $\theta_l$。

（2）输出层（隐结点到输出结点间）的修正公式

① 输出结点的样本实际输出：$t_l$

② 误差控制：

所有样本误差：$E = \sum_{k=1}^{P} e_k < \varepsilon$，其中一个样本误差

$$e_k = \sum_{l=1}^{n} \left| t_l^{(k)} - O_l^{(k)} \right|$$

其中，$p$ 为样本数，$n$ 为输出结点数。

③ 误差公式：

$$\delta_l = (t_l - O_l) \cdot O_l \cdot (1-O_l) \tag{9.37}$$

④ 权值修正：

$$T_{li}(k+1) = T_{li}(k) + \eta \delta_l y_i \tag{9.38}$$

其中 $k$ 为迭代次数。

⑤ 阈值修正：

$$\theta_l(k+1) = \theta_l(k) + \eta \delta_l \tag{9.39}$$

（3）隐结点层（输入结点到隐结点间）的修正公式

① 误差公式：

$$\delta'_i = y_i(1-y_i) \sum_l \delta_l T_{li} \tag{9.40}$$

② 权值修正：

$$W_{ij}(k+1) = W_{ij}(k) + \eta' \delta'_i x_j \tag{9.41}$$

③ 阈值修正：

$$\theta_i(k+1) = \theta_i(k) + \eta' \delta'_i \tag{9.42}$$

**8. BP 模型算法总结**

BP 模型算法分为三个部分：①隐结点和输出结点的输出计算；②输出结点和隐结点的误差计算；③输出层网络权值及结点阈值与隐结点层网络权值及结点阈值的修改，如图 9.8 所示。

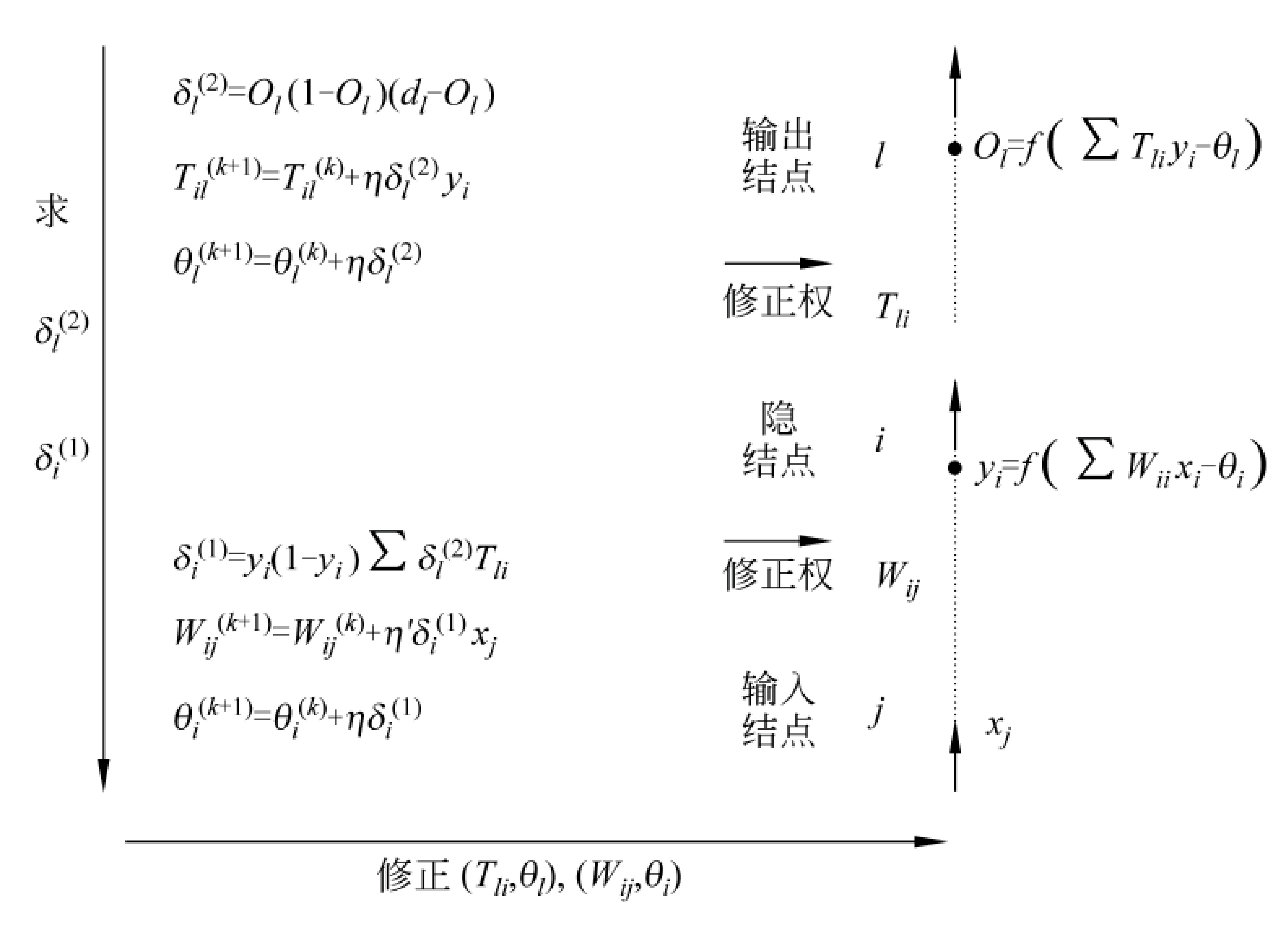


图 9.8 BP 模型算法示意图

BP 模型计算，不但对每一个样本要积累计算各输出结点的误差，对所有样本还要积累各样本的误差，这个总误差才是一次迭代的误差，当它不满足给定误差时，继续迭代(用新网络权值和阈值，再对所有样本重复计算)，直到满足给定误差为止。这种迭代可能需要上万次才能够收敛。

## 9.2.3 BP 网络的典型实例

**1. 异或问题的 BP 神经网络**

异或问题(XOR)用 BP 模型进行求解，样本和神经网络如图 9.9 所示。

按问题要求，设置输入结点为两个($x_1$，$x_2$)，输出结点为 1 个($z$)，隐结点定为 2 个($y_1$，$y_2$)。

| 输入 | $x_1$ | $x_2$ | 输出 |
|---|---|---|---|
| | 0 | 0 | 0 |
| | 0 | 1 | 1 |
| | 1 | 0 | 1 |
| | 1 | 1 | 0 |

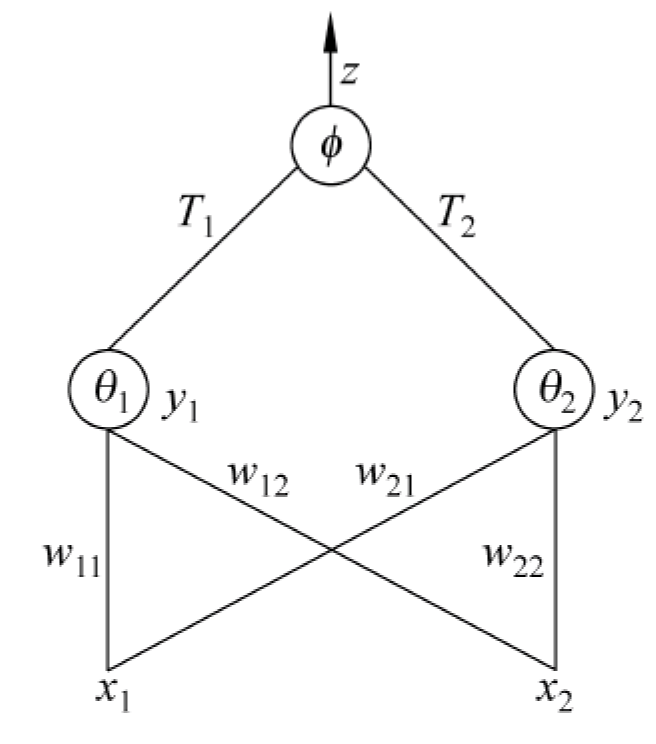


图 9.9 异或问题神经网络图

**2. 计算机运行结果**

(1) 迭代次数：16745 次；给定误差：0.05

(2) 隐层网络权值和阈值：

$w_{11}=5.24,\quad w_{12}=5.23,\quad w_{21}=6.68,\quad w_{22}=6.64,\quad \theta_1=8.01,\quad \theta_2=2.98$

(3) 输出层网络权值和阈值：

$$T_1=-10,\quad T_2=10,\quad \phi=4.79$$

# 9.3 径向基函数网络

## 9.3.1 径向基函数 RBF 网络原理

### 1. 基本概念

径向基函数(Radial Basis Function,RBF)神经网络是一类常用的三层前馈网络,用于模式分类,也可用于函数逼近。

RBF 神经网络与前向神经网络类似,是含有隐层的前向三层神经网络。输入层由一些输入单元组成。隐层单元的变换函数是径向基函数,它是非线性的,在输入空间到隐层空间之间进行非线性变换。输出层的输出是对隐层单元的线性分类,即 RBF 神经网络分成前后两部分：

(1) 将非线性样本：$X(x_1,x_2,\cdots,x_n)$,通过径向基函数变换成线性样本：$H(h_1(x),h_2(x),\cdots,h_k(x))$。

(2) 通过类似于感知机神经网络,将线性样本 $H(X)$进行分类。

径向基函数神经网络结构图如图 9.10 所示。

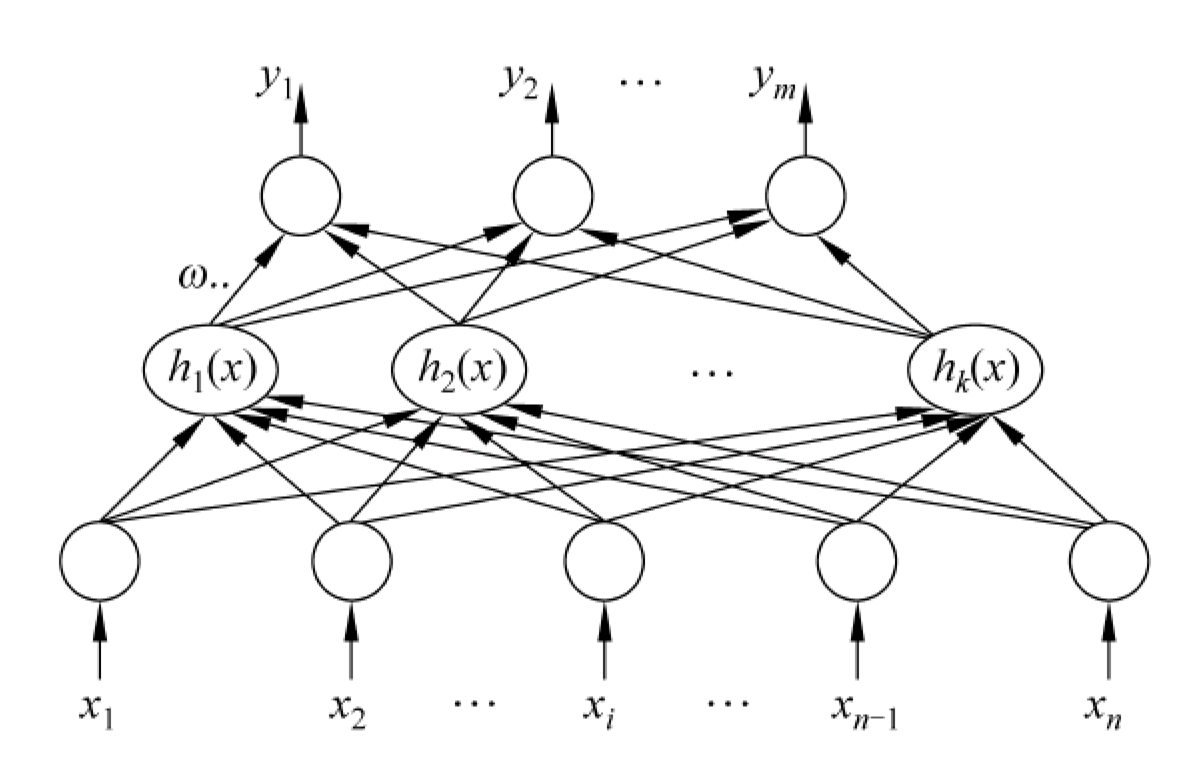



图 9.10　RBF 径向基函数神经网络结构图

径向基函数神经网络克服了反向传播网络(BP)梯度下降算法中的局部极小问题。其中,隐单元的个数可以是固定的,也可以根据对象的特征自适适应选择。当隐单元的个数和径向基函数的参数确定后,RBF 神经网络的训练学习只集中在隐层至输出层的连接权值的训练学习,这时的训练是基于线性寻优的。RBF 网络具有逼近精度高、网络规模小、学习速度快和不存在局部最小问题等特点。

径向基函数取高斯函数,即隐层第 $k$ 个结点的输出为

$$h_k(x)=\exp\left(-\frac{\|x-c_k\|}{2\sigma_k^2}\right)$$

式中 $c_k=(c_{1k},c_{2k},\cdots,c_{nk})$为第 $k$ 个隐结点的中心向量;$\sigma_k$ 为第 $k$ 个隐结点的宽度;$\|*\|$ 为输入向量 $x(x_1,x_2,\cdots,x_n)$与中心向量 $c_k$的距离。

整个网络的输入输出方程为

$$y_i=\sum_{j=1}^{k}\omega_{ij}h_j(x),\quad 1,2,\cdots,m$$

式中:$k$ 为当前网络中隐结点的个数;$\omega_{ij}$ 为隐层第 $j$ 个终结点与输出层第 $i$ 个结点的连接权值。由于输出层是线性函数,网络输出是径向基隐层输出的线性组合,因此很容易达到从输入空间到输出空间非线性映射的目的。

#### 2. 隐层设计

设计 RBF 网络的隐层,主要是确定隐层单元数目和它们的激励函数(高斯函数),高斯函数由聚类中心 $c_k$ 和聚类宽度 $\sigma_k$ 确定。在初始计算时,由于训练模式的类别是已知的,因此可以采用 K-均值聚类方法,即每一类对应隐层网络的一个单元。高斯函数的宽度参数可以根据各类中的点与中心点的距离取均值,而高斯函数的中心分别取各类的均值作为各自单元的中心点。虽然这种方法构造的网络相对显得有点粗糙,但是通过有效的学习算法以及采用误差校正的策略可以有效地改善网络的性能。在实际分类应用中,通过这样的方式构造的网络相当简洁高效,而且不会影响分类的准确性。

### 9.3.2 RBF 网络算法与分析

#### 1. RBF 网络的训练

假设有一个样本序列$\{x(l),d(l)\}$,其中 $x(l)$第 $l$ 个输入样本,$d(l)$为 $x(l)$对应的实际输出,这些样本一个接一个地提供给网络。在某时刻 $t$ 输入样本$(x(t),d(t))$,根据网络计算输出与样本实际输出的差异程度来决定是否需要修改网络参数。

常用的训练算法,包括聚类方法,梯度训练算法,以及正交最小二乘学习算法。下面介绍常用梯度训练算法。

#### 2. 梯度训练算法公式

RBF 网络的梯度训练方法与 BP 算法训练多层感知器的原理类似,也是通过最小化目标函数实现对各隐结点数据中心、宽度和输出权值的学习。

建立这种训练过程首先是定义一个误差函数 $E$ 来衡量网络计算输出 $y$ 与样本实际输出 $d$ 之间的差异,形式为

$$E=\frac{1}{2}\sum_{i=1}^{m}(d_i-y_i)^2$$

这里的 $d_i$,$y_i$ 分别为网络第 $i$ 个输出结点的实际输出和计算输出。按照梯度下降方法,和 BP 算法一样,用误差函数 $E$ 分别对 $\omega_{ij}$、$c_k$、$\sigma_k$ 求偏导数,得出此三个数的迭代公式:

$$\omega_{ij}^{t+1}=\omega_{ij}^{t}+\eta_\omega\cdot(d_i-y_i)\cdot h_j(x)$$

$$C_j^{t+1}=C_j^t+\eta_c\sum_{i=1}^{m}[(d_i-y_i)\cdot\omega_{ij}]\cdot\frac{(x-c_i)}{\sigma_j^2}\cdot h_j(x)$$

$$\sigma_j^{t+1} = \sigma_j^t + \eta_\sigma \sum_{i=1}^{m} [(d_i - y_i) \cdot \omega_{ij}] \cdot \frac{\| x - c_j \|^2}{\sigma_j^3} \cdot h_j(x)$$

根据上述公式修改网络参数，经过有限次的调整之后，就可以使该样本的网络输出误差控制在容许的误差范围以内。

#### 3. 基本的 RBF 算法的流程

(1) 根据已知样本的类别个数确定隐层结点数目 $k$，并分别在各类别中随机选取一个样本作为该类的中心 $c_k$，初始宽度 $\sigma_k$ 可以通过各类中的样本标准差确定，再初始化 $\omega_{ij}$。设定容许误差 $\varepsilon(\varepsilon \geqslant 0)$、学习率 $\eta_\omega, \eta_c, \eta_\sigma$。假设目前可用的最大训练样本数为 $S(S \geqslant 1)$，设定循环变量 $t$ 初始为 1。

(2) 输入第 $t$ 个训练样本，按公式求得网络实际输出 $y$。

(3) 求计算输出与实际输出之间的误差 $E$；如果误差为 $|E| \leqslant \varepsilon$，则该样本不需要调整网络参数，跳到(6)；否则进入下一步。

(4) 按 $i=1,2,\cdots,m; j=1,2,\cdots,k$，对权值 $\omega_{ij}$、中心矢量 $c_k$、宽度 $\sigma_k$ 进行修正。

(5) 基于新的网络参数($t+1$)，转到(2)。

(6) $t+1$；如果 $t \geqslant S$(样本数)，即无新样本，则整个学习过程结束；否则转到(2)。

#### 4. RBF 神经网络算法分析

径向基函数(RBF)神经网络是一种三层前馈神经网络，其隐含层和输出层所完成的功能是不同的。隐含层是对 RBF 的输入参数进行调整，采用的是非线性样本转换成线性样本的策略。输出层是对线性样本进行分类，采用的是线性优化策略。这样，其网络训练算法包括两部分：

(1) 网络隐层的径向基函数的中心 $c_j$ 和宽度 $\sigma_j$ 的确定；

(2) 隐层到输出层权值 $\omega_{ij}$ 的确定，即调整隐层空间到输出空间的权值矩阵。

这两个部分可以同步进行(上面给出的 RBF 算法流程)，也可以异步进行(对第(1)步进行训练样本的聚类变换，对第(2)步进行类似于感知机神经网络计算，完成分类)。

因为网络输出与连接权之间呈线性关系，所以对权值的训练可以采用类似于感知机神经网络算法。因此，RBF 神经网络训练算法的关键在于第(1)步的径向基函数的中心矢量和宽度的确定。

## 9.4 神经网络的几何意义

### 9.4.1 神经网络的超平面含义

#### 1. 神经元与超平面

由 $n$ 个神经元($j=1,2,\cdots,n$)对连接于神经元 $i$ 的信息总输入 $I_i$ 为

$$I_i = \sum_{j=1}^{n} w_{ij} x_j - \theta_i \tag{9.43}$$

其中 $w_{ij}$ 为神经元 $j$ 到神经元 $i$ 的连接权值，$\theta_i$ 为神经元的阈值。神经元 $x_j(j=1,2,\cdots,n)$ 相当于 $n$ 维空间$(x_1,x_2,\cdots,x_n)$中一个结点的坐标(为了便于讨论，省略 $i$ 下标记)。令：

$$I=\sum_{j=1}^{n}w_jx_j-\theta=0 \tag{9.44}$$

它代表了 $n$ 维空间中，以坐标 $x_j$ 为变量的一个超平面。其中 $w_j$ 为坐标的系数，$\theta$ 为常数项。

若已知有 $n$ 个样本：

$$(x_1^{(k)},x_2^{(k)},\cdots,x_n^{(k)})\quad k=1,2,\cdots,n$$

在 $n$ 维空间中，相当于已知 $n$ 个结点的各结点坐标。该 $n$ 个结点可唯一构成一个超平面。超平面方程用行列式表示为

$$\begin{vmatrix} x_1 & x_2 & \cdots & x_n & 1 \\ x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} & 1 \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{(n)} & x_2^{(n)} & \cdots & x_n^{(n)} & 1 \end{vmatrix}=0 \tag{9.45}$$

它是以 $n$ 维坐标 $x_j(j=1,2,\cdots,n)$为变量的线性方程。将它展开即为超平面方程(9.44)。

其中系数 $w_j$ 和常数 $\theta$ 用行列式表示为

$$w_j=(-1)^{1+j}\begin{vmatrix} x_1^{(1)} & \cdots & x_{j-1}^{(1)} & x_{j+1}^{(1)} & \cdots & x_n^{(1)} & 1 \\ x_1^{(2)} & \cdots & x_{j-1}^{(2)} & x_{j+1}^{(2)} & \cdots & x_n^{(2)} & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{(n)} & \cdots & x_{j-1}^{(n)} & x_{j+1}^{(n)} & \cdots & x_n^{(n)} & 1 \end{vmatrix} \tag{9.46}$$

$$-\theta=(-1)^n\begin{vmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(i)} & x_2^{(i)} & \cdots & x_n^{(i)} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(n)} & x_2^{(n)} & \cdots & x_n^{(n)} \end{vmatrix} \tag{9.47}$$

当 $n=2$ 时，“超平面”为平面$(x_1,x_2)$上的一条直线：

$$I=\sum_{j=1}^{2}w_jx_j-\theta=w_1x_1+w_2x_2-\theta=0$$

当 $n=3$ 时，“超平面”为空间$(x_1,x_2,x_3)$上的一个平面：

$$I=\sum_{j=1}^{3}w_jx_j-\theta=w_1x_1+w_2x_2+w_3x_3-\theta=0$$

从几何角度看，一个神经元代表一个超平面。

**2. 超平面的作用**

$n$ 维空间$(x_1,x_2,\cdots,x_n)$上的超平面 $I=0$，将空间划分为三部分。

(1) 平面本身

超平面上的任意结点$(x_1^{(0)},x_2^{(0)},\cdots,x_n^{(0)})$满足于超平面方程，即：

$$\sum_j w_j x_j^{(0)} - \theta = 0 \tag{9.48}$$

（2）超平面上部 $P$

超平面上部 $P$ 的任意结点（$x_1^{(p)}, x_2^{(p)}, \cdots, x_n^{(p)}$）满足于不等式，即

$$\sum_j w_j x_j^{(p)} - \theta > 0 \tag{9.49}$$

（3）超平面下部 $Q$

超平面下部 $Q$ 的任意结点（$x_1^{(q)}, x_2^{(q)}, \cdots, x_n^{(q)}$）满足于不等式，即

$$\sum_j w_j x_j^{(q)} - \theta < 0 \tag{9.50}$$

### 3. 作用函数的几何意义

神经网络中使用的阶梯形作用函数为

$$f(x) = \begin{cases} 1, & x > 0 \\ 0, & x \leqslant 0 \end{cases}$$

把 $n$ 维空间中超平面的作用和神经网络作用函数结合起来，即

$$f(I) = f\left(\sum w_j x_j - \theta\right) = \begin{cases} 1, & \sum\limits_j w_j x_j - \theta > 0 \\ 0, & \sum\limits_j w_j x_j - \theta \leqslant 0 \end{cases} \tag{9.51}$$

它的含义为：超平面上部 $P$ 的任意结点经过作用函数后转换成数值 1。超平面上任意结点和超平面下部 $Q$ 上的任意结点经过作用函数后转换成数值 0。

### 4. 神经元的几何意义

通过以上分析可知，一个神经元将其他神经元对它的信息总输入 $I$，作用以后（通过作用函数）的输出，相当于该神经元所代表的超平面将 $n$ 维空间（$n$ 个输入神经元构成的空间）中超平面上部结点 $P$ 转换成 1 类，超平面及其下部结点转换成 0 类。

结论：神经元起了一个分类作用。

### 5. 线性样本与非线性样本

定义：对空间中的一组两类样本，当能找出一个超平面将两者分开，称该样本是线性样本。若不能找到一个超平面将两者分开，则称该样本是非线性样本。

二值逻辑加法样本示意图如图 9.11 所示，两类样本（0，1）可以利用一条直线分隔开。

从线性样本定义可知二值逻辑加法是线性可分的。

感知机对线性样本是非常有效的，它在模式识别中是一个重要的方法。

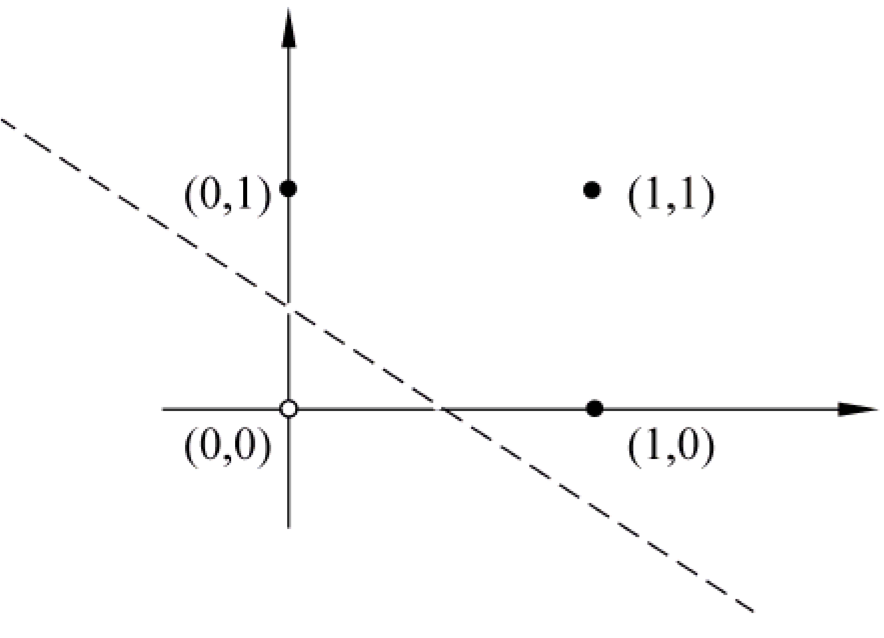


图 9.11　二值逻辑加法样本示意图

从图 9.5 中可以看出，异或问题（XOR）是找不

到一条直线将两类样本分开。从线性样本定义可知,异或问题样本是一个非线性样本。

#### 6. 非线性样本变换成线性样本

利用超平面分割空间原理,对一个非线性样本它是不能用一个超平面分割开。但用多个超平面分割空间成若干区,使每个区中只含同类样本的结点。这种分割完成了一种变换,使原非线性样本变换成二进制值下的新线性样本。

### 9.4.2 异或问题的实例分析

异或问题(XOR)的解已在9.2.3节中给出,根据神经网络的几何意义来进行具体分析。

#### 1. 隐结点代表的直线方程

如图9.12所示,利用隐结点的权值和阈值可以建立隐结点代表的直线方程:

$$y_1: 5.24x_1 + 5.23x_2 - 8.01 = 0$$

即

$$x_1 + 0.998x_2 - 1.529 = 0 \qquad (9.52)$$

$$y_2: 6.68x_1 + 6.64x_2 - 2.98 = 0$$

即

$$x_1 + 0.994x_2 - 0.446 = 0 \qquad (9.53)$$

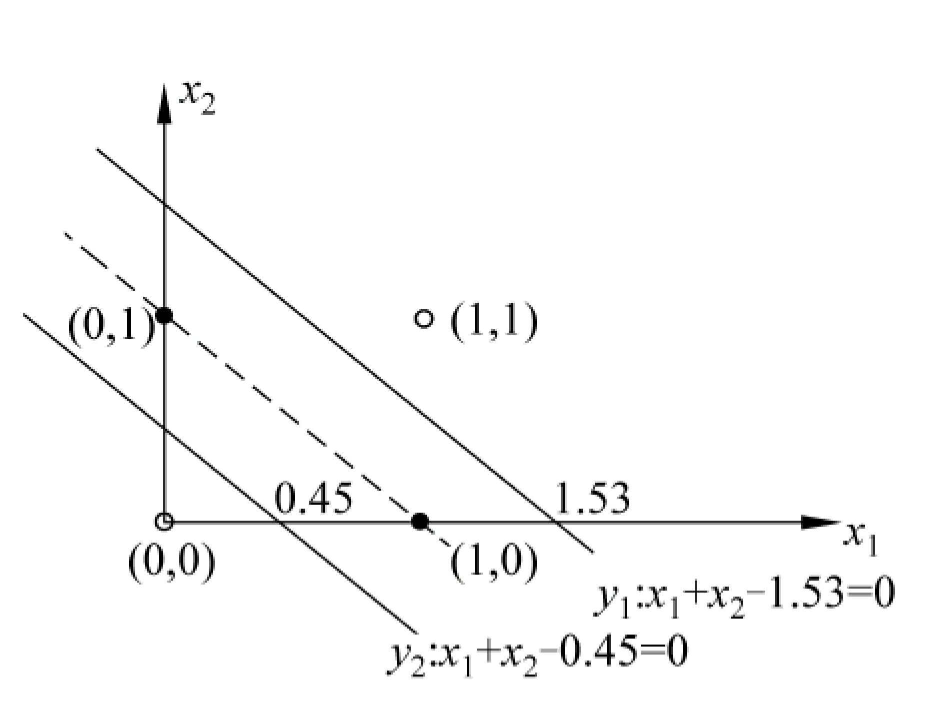



图9.12 隐结点代表的直线方程

直线 $y_1$ 和 $y_2$ 将平面 $(x_1, x_2)$ 分为三区:

- $y_1$ 线上方区,$x_1+x_2-1.53>0$,$x_1+x_2-0.45>0$
- $y_1$,$y_2$ 线之间区,$x_1+x_2-1.53<0$,$x_1+x_2-0.45>0$
- $y_2$ 线的下方区,$x_1+x_2-1.53<0$,$x_1+x_2-0.45<0$

对样本点:

- 点(0,0)落入 $y_2$ 线的下方区,经过隐结点作用函数 $f(x)$(暂取它为阶梯函数),得到输出 $y_1=0$,$y_2=0$。
- 点(1,0)和点(0,1)落入 $y_1$,$y_2$ 线之间区,经过隐结点作用函数 $f(x)$,得到输出均为 $y_1=0$,$y_2=1$。
- 点(1,1)落入 $y_1$ 线上方区,经过隐结点作用函数 $f(x)$,得到输出为 $y_1=1$,$y_2=1$。

结论:隐结点将 $x_1$,$x_2$ 平面上四个样本点(0,0),(0,1),(1,0),(1,1)变换成三个样本点(0,0),(0,1),(1,1),它已是线性样本。

#### 2. 输出结点代表的直线方程

如图9.13所示,利用输出结点的权值和阈值可以建立隐结点代表的直线方程:

$$Z: -10y_1 + 10y_2 - 4.79 = 0$$

即

$$-y_1 + y_2 - 0.479 = 0 \qquad (9.54)$$

直线 $Z$ 将平面$(y_1, y_2)$分为两区：

- $Z$ 线上方区：$-y_1+y_2-0.479>0$；
- $Z$ 线下方区：$-y_1+y_2-0.479<0$。

对样本点：

- 点(0,1)(即 $y_1=0, y_2=1$)落入 $Z$ 线上方区，经过输出结点作用函数 $f(x)$(暂取它为阶梯函数)得到输出为：$Z=1$；
- 点(0,0)(即 $y_1=0, y_2=0$)，点(1,1)(即 $y_1=1, y_2=1$)落入 $Z$ 线下方区，经过输出结点作用函数 $f(x)$得到输出为：$Z=0$。

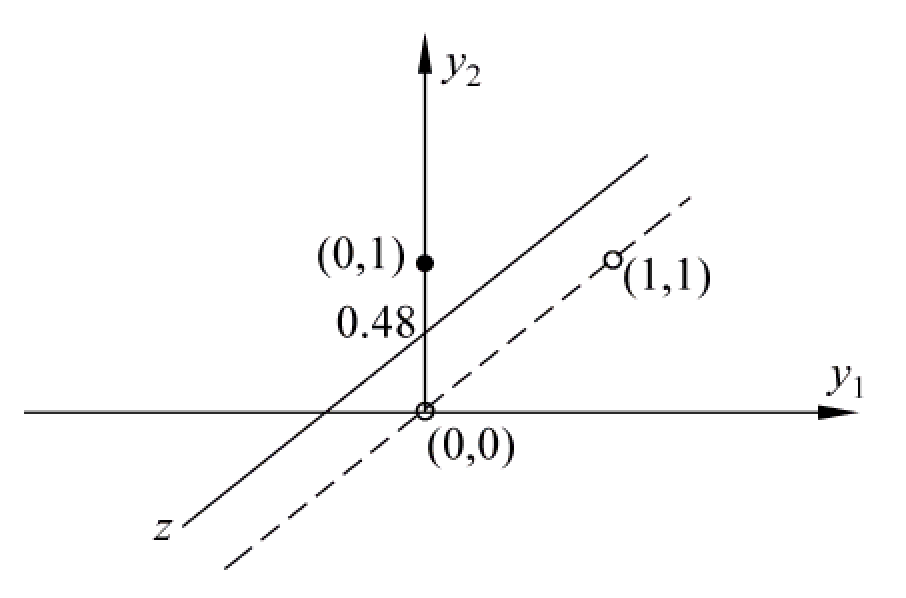


图 9.13　输出结点代表的直线方程

结论：输出结点将 $y_1, y_2$ 平面上三个样本(0,0),(0,1),(1,1)变换成两类样本 $Z=1$ 和 $Z=0$。

**3. 神经网络结点的作用**

从上面的分析中可以得出结论：

① 隐结点作用是将原非线性样本(四个)变换成线性样本(三个)。

② 输出结点作用是将线性样本(三个)变换成两类(1 类或 0 类)。

对于作用函数 $f(x)$取为 S 型函数，最后变换成两类为"接近 1 类"和"接近 0 类"。

**4. 超平面(直线)特性**

(1) 隐结点直线特性

隐结点直线 $y_1, y_2$ 相互平行，且平行于过(1,0)点和(0,1)的直线 $L$：$x_1+x_2-1=0$。

直线 $y_1$ 位于点(1,1)到直线 $L$ 的中间位置附近($\theta_1=1.53$)。

直线 $y_2$ 位于点(0,0)到直线 $L$ 的中间位置附近($\theta_2=0.45$)。

阈值 $\theta_1$ 和 $\theta_2$ 可以在一定范围内变化：$1.0\leqslant\theta_1<2, 0\leqslant\theta_2<1.0$。其分类效果是相同的。这说明神经网络的解(网络权值和阈值)可以是多个(即多条不同的直线)。

(2) 输出结点直线特性

输出结点直线 $Z$，平行于过点(0,0)和点(1,1)直线 $P$：$y_1-y_2=0$。

直线 $Z$ 位于点(0,1)到直线 $P$ 的中间位置附近($\phi=0.48$)。

阈值 $\phi$ 可以在一定范围内变化($0\leqslant\phi<1$)，其分类效果是相同的，输出层的权值和阈值也是多解。

**5. 超曲面神经网络概念**

超曲面神经网络是相对于超平面神经网络而言的。传统的神经网络是以 MP 模型为基础的，按 MP 模型，神经网络的公式为

$$O_i = f\left(\sum_j w_{ij} x_j - \theta_j\right) \quad i = 1, 2, \cdots, n \tag{9.55}$$

表示每个神经网络元 $O_i$ 的输入 $I_i$ 代表了一个超平面(其中 $x_j$ 是一次方)，即：

$$I_i=\sum_j w_{ij}x_j-\theta_j=0 \tag{9.56}$$

神经网络的作用函数：

$$f(I_i)=f(\sum_j w_{ij}x_j-\theta_j)=\begin{cases}1, & \sum w_{ij}x_j-\theta_j>0\\ 0, & \sum w_{ij}x_j-\theta_j\leqslant 0\end{cases} \tag{9.57}$$

相当于超平面 $I_i$ 对 $n$ 维空间进行了一次分割。多个超平面 $I_i(i=1,2,\cdots,n)$对 $n$ 维空间进行了组合分割，把 $n$ 维空间分成了若干个区域，使每个区域中，只包含同类样本。这种区域分割完成了一次变换，即将非线性样本（不能用一个超平面分割的样本）通过多个超平面的分割使它变成了线性样本。对新的线性样本，再通过一次神经网络（超平面）就可完成对它的分割（分类）。BP 神经网络模型实质上就是通过两次超平面分割（即隐结点层和输出结点层）来完成样本分类的。

BP 神经网络是反复通过神经网络修改权值的迭代，最后找出隐结点神经网络超平面和输出结点神经网络超平面。

除了用超平面分割空间外，能否用超曲面分割空间实现对非线性样本的分割呢？这就要求神经网络公式(9.55)中 $x_j$ 变成二次方以上。

黄金才提出的“超圆神经网络模型 CC”的公式：

$$y=f\left(\sum_i(x_i-a_i)^2-c^2\right) \tag{9.58}$$

该神经网络模型与 MP 神经网络模型的比较，如图 9.14 所示。

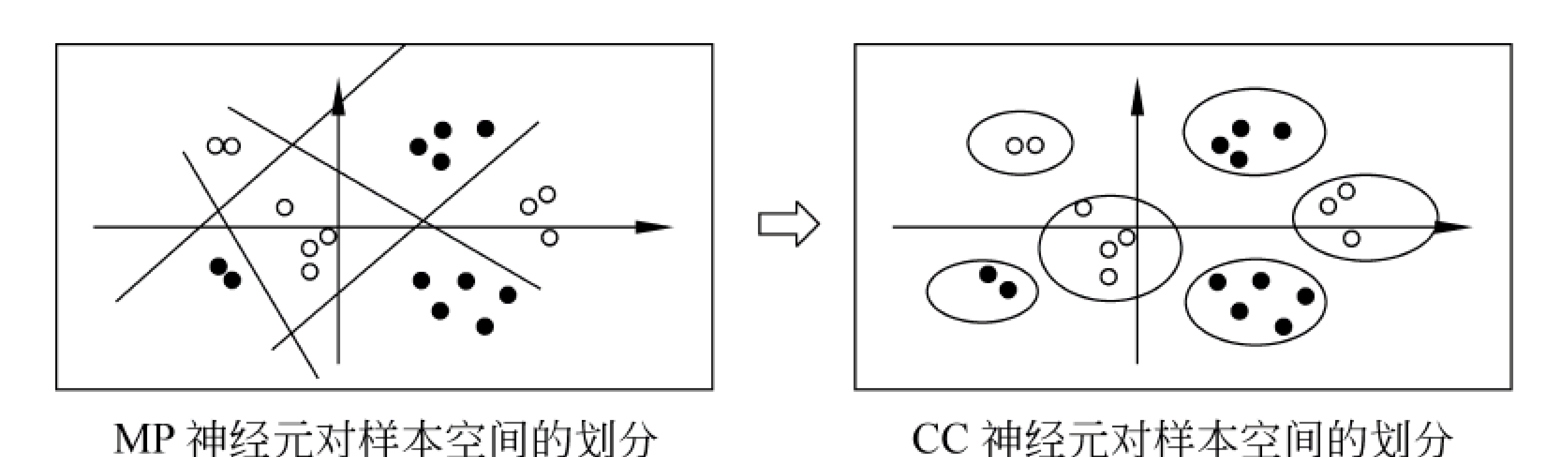



图 9.14　CC 模型与 MP 模型对样本空间的划分比较

他还提出了“超曲面神经网络模型 Cover”的公式：

$$y=f(w_1x+w_2y+w_3x^2+w_4xy+w_5y^2-c) \tag{9.59}$$

以上超曲面神经网络有效地达到了对非线性样本的分类效果。超曲面神经网络是对神经网络的有益扩展。

近几年发展起来的支持向量机（support vector machines）是在统计学习理论的基础上直接构造这些超平面和超曲面函数（与神经网络无关），来完成对线性和非线性各类样本的分割，这项工作已经取得了很大的成果。这样看来，它已经不属于神经网络了。

# 习　题　9

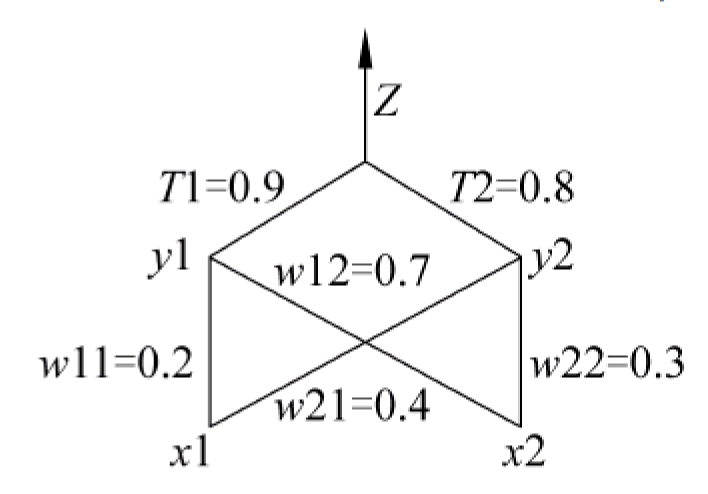
1. 说明神经网络的 MP 模型原理。

2. 对比感知机权值修正公式和 Hebb 规则的含义。

3．说明阶梯函数与S型函数的相同与不同，并说明它们的作用。

4．BP模型中误差公式 $\delta_i = f'(\text{net}_i)\sum_k \delta_k \cdot w_{ki}$ 的含义是什么？

5．对如下BP神经网络写出它的计算公式（含学习公式），并对其初始权值以及样本 $x1=1, x2=0, d=0$ 进行一次神经网络计算和学习（该系数 $\eta=1$，各点阈值为0）。




作用函数简化为

$$y = f(x) = \begin{cases} 0.95, & x \geqslant 0.45 \\ x + 0.5, & -0.45 < x < 0.45 \\ 0.05, & x \leqslant -0.45 \end{cases}$$

6．编制BP网络模型程序，完成异或问题的计算。

7．说明径向基函数网络的原理。

8．比较RBF网络与BP网络在修正权值上的异同。

9．神经元网络的几何意义是什么？

10．说明下列样本是什么类型样本，为什么？

(1)

| 输　　入 | | 输　　出 |
|---|---|---|
| $x1$ | $x2$ | $d$ |
| 0 | 0 | 0 |
| 0.5 | 0.5 | 1 |
| 1 | 1 | 0 |

(2)

| 输　　入 | | 输　　出 |
|---|---|---|
| $x1$ | $x2$ | $d$ |
| 0 | 0 | 0 |
| 0.5 | 0 | 1 |
| 1 | 1 | 0 |

# 第10章　遗传算法与进化计算

## 10.1　遗传算法

遗传算法是模拟生物进化的自然选择和遗传机制，将其转换成数学形式的遗传算子，通过迭代（遗传）计算形成了一种寻优算法。它模拟了生物的繁殖、交配和变异现象，形成了选择、交叉、变异三个算子。从任意一初始种群出发（问题的初始解），产生一群新的更适应环境的后代（问题的新解）。这样一代一代不断繁殖、进化（迭代），最后收敛到一个最适应环境的个体上（问题的最终解）。遗传算法对于复杂的优化问题，无需建立像运筹学中的数学模型并进行复杂运算，只需要利用遗传算法的算子就能寻找到问题的最优解或满意解。

自然选择学说认为，生物要生存下去，就必须进行生存斗争。生存斗争包括种内斗争、种间斗争以及生物跟环境之间的斗争三个方面。在生存斗争中，具有有利变异的个体容易存活下来，并且有更多的机会将有利变异传给后代；具有不利变异的个体就容易被淘汰，产生后代的机会也少得多。因此，凡是在生存斗争中获胜的个体都是对环境适应性比较强的。

达尔文把这种在生存斗争中“适者生存、不适者淘汰”的过程叫做自然选择。自然选择学说表明，遗传和变异是决定生物进化的内在因素。遗传是指父代与子代之间，在性状上存在的相似现象。变异是指父代与子代之间，以及子代个体之间，在性状上或多或少地存在的差异现象。在生物体内，遗传和变异的关系十分密切。一个生物体的遗传性状往往会发生变异，而变异的性状有的可以遗传。遗传能使生物的性状不断地传送给后代，因此保持了物种的特性，变异能够使生物的性状发生改变，从而适应新的环境而不断地向前发展。

生物的遗传与变异有它的物质基础。遗传物质的主要载体是染色体（Chromsome）。在遗传算法中称为个体，它是数学问题的解（初始解、中间解、最终解）。染色体主要是由DNA（脱氧核糖核酸）和蛋白质组成的，基因（Gene）是染色体的片段，它储存着遗传信息，可以准确地复制，也能够发生突变，生物体自身通过对基因的复制（Reproduction）和交叉（Crossover，即基因自由组合和基因连锁互换）的操作实现性状的遗传。在遗传算法中的个体是由数学问题的参数组成，通过三个遗传算子的迭代求出问题的最优解。

### 10.1.1　遗传算法基本原理

**1. 概述**

遗传算法（Genetic Algorithms，GA）是一种基于遗传学的搜索优化算法。遗传学认为遗传是作为一种指令码封装在每个染色体个体中，并以基因（位）的形式包含在染色体（个体）中。每个基因有特殊的位置并控制某个特殊的性质，由基因组成的个体对环境有一定的适应性。基因杂交和基因突变能产生对环境适应性强的后代，通过优胜劣汰的自然选择，适应值高的基因结构就保存了下来。

在遗传算法中,“染色体”对应的是问题的解,通常是由一维串结构的数据(问题的参数的组合)来表现的。串上各个位置对应“基因”(每个参数),而各位置上的值对应基因的取值。基因组成的串就是染色体,或者叫做基因型个体(Individuals)。一定数量的个体组成了群体(Population)。群体中个体的数目称为群体的大小(Population size),也叫群体规模。而各个体对环境的适应程度叫做适应度(Fitness)。

遗传算法中包含两个必须的数据转换操作,一个是把搜索空间中数学问题参数的组合的解转换成遗传空间中的个体(染色体),此过程又叫做编码(Coding)操作;另一个是相反的操作,叫做译码(Decoding)操作。

遗传算法是一种群体型操作,该操作以群体中的所有个体为对象。选择(Selection)、交叉(Crossover)和变异(Mutation)是遗传算法的三个主要操作算子,它们构成了遗传操作(Genetic Operation),使遗传算法具有其他传统方法所没有的特性。

遗传算法的处理流程如图 10.1 所示。

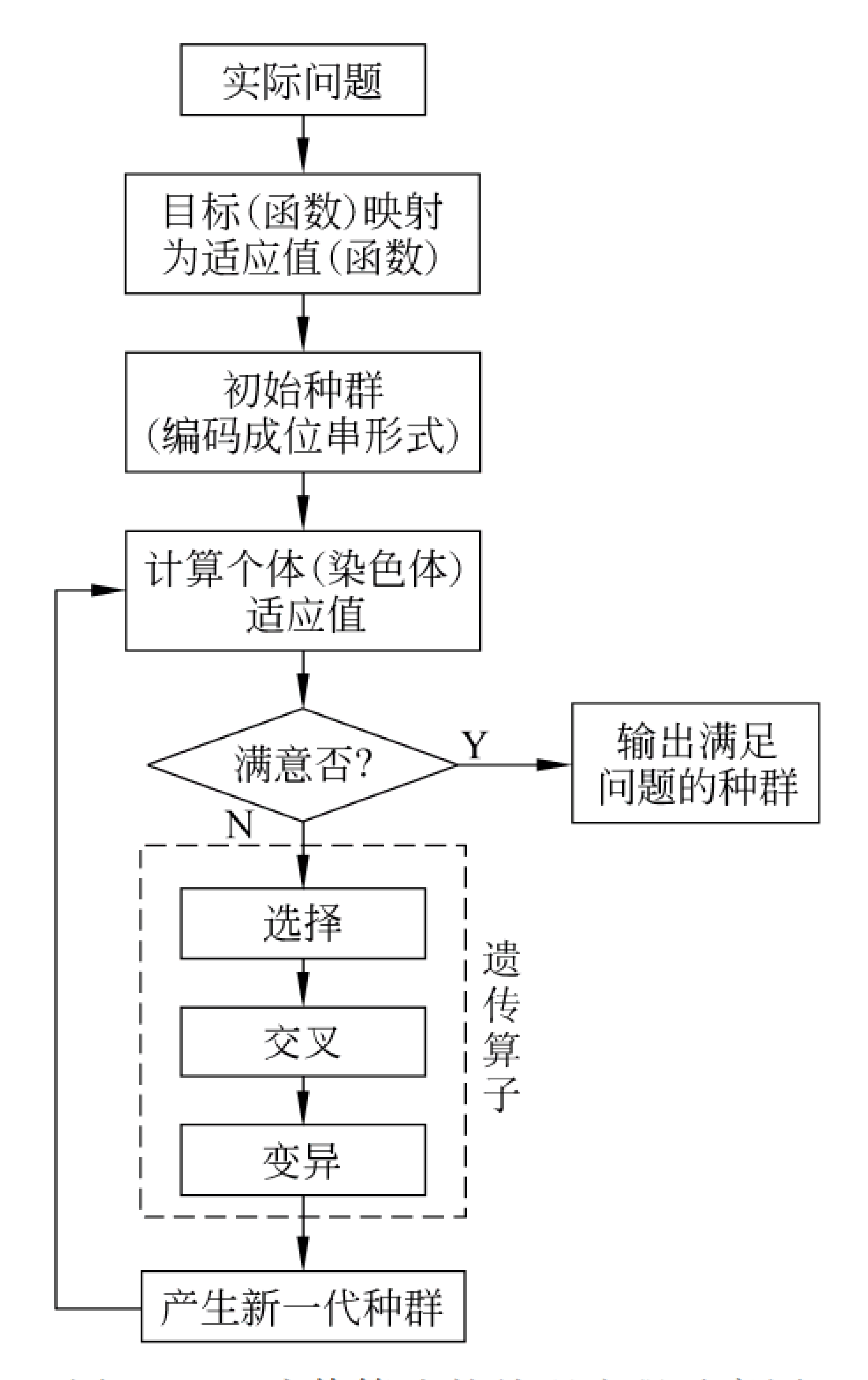



图 10.1　遗传算法的处理流程示意图

遗传算法首先将问题的每个可能的解按某种形式进行编码,编码后的解称为染色体(个体)。随机选取 $N$ 个染色体构成初始种群,再根据预定的评价函数对每个染色体计算适应值,使得性能较好的染色体具有较高的适应值。选择适应值高的染色体进行复制,通过遗传算子:选择、交叉(重组)、变异,来产生一群新的更适应环境的染色体,形成新的种群。这样一代一代不断繁殖、进化,最后收敛到一个最适应环境的个体上,求得问题的最优解。

**2. 遗传算法中的基本要素**

遗传算法中包含了如下五个基本要素:①问题编码;②初始群体的设定;③适应值函数的设计;④遗传操作设计;⑤控制参数设定(主要是指群体大小和使用遗传操作的概率等)。这五个要素构成了遗传算法的核心内容。

(1) 问题编码

将子串拼接起来构成“染色体”位串。但是不同串长和不同的码制,对问题求解的精度和遗传算法收敛时间会有很大影响。如何将问题描述成串的形式就不那么简单,而且同一问题可以有不同的编码方法。

常用的二进制编码方式是基于确定的二进制位串上:$I=\{0,1\}^L$。目前也出现了其他编码方式,如用向量(向量元素为实数)来表示染色体,或者用规则形式(规则 A,规则 B,规则 C,…)来表示染色体。

(2) 初始群体的生成

遗传算法是群体型操作,这样必须为遗传操作准备一个由若干初始解组成的初始群体。

初始群体的每个个体都是通过随机方法产生的。初始群体也称作为进化的初始代，即第一代(first generation)。

(3) 适应值函数的确定

遗传算法在搜索进化过程中一般不需要其他外部信息，仅用评价函数值来评价个体或解的优劣，并作为以后遗传操作的依据。评价函数值又称做适应值(fitness)。

适应值函数(即评价函数)是根据目标函数确定的。适应值总是非负的，任何情况下总是希望越大越好。一般目标函数有正有负，且和适应值之间的关系也是多种多样的。例如求最大值时，目标函数与适应值变化方向一致，而求最小值时，变化方向正好相反。因此，存在目标函数到适应值函数的映射问题，常见的映射形式为

$$\phi(\alpha) = \delta(f(\tau(\alpha)))$$

其中，$\alpha$ 为个体；$\tau(\alpha)$为个体的译码函数；$f$ 则为具体求解问题的表达式；$\delta$ 为变换函数，$\delta$ 的作用是确保适应值为正，并且最好的个体其适应值最大。适应值函数的选取至关重要，它直接影响到算法的收敛速度即最终能否找到最优解。函数优化问题可直接将函数本身作为评价函数。而对于复杂系统的评价函数一般不那么直观，往往需要研究者自己构造出能对解的性能进行评价的函数。

为了使遗传算法有效地工作，必须保持种群内位串的多样性和位串之间的竞争机制。如果将遗传算法的运行分为开始、中间和结束三个阶段，在开始阶段，若一个规模不太大的种群内有少数非凡的个体(适应值很高的位串)的话，按通常的选择方法，这些个体会被大量繁殖，在种群中占有较大的比重，这样就会减少种群的多样性，导致过早收敛，从而可能丢失一些有意义的搜索点或最优点，而陷入局部最优。其次，在结束阶段，即使种群内保持了很大的多样性，但若所有或大多数个体都有很高的适应值，从而种群平均适应值和最大适应值相差无几，那么平均适应值附近的个体和具有最高适应值的个体被选中的机会几乎相同，这样选择就成了一个近乎随机的步骤，适应值的作用就会消失，从而搜索性能得不到明显改进。因此，有必要对种群内各位串的适应值进行有效的调整，既不能相差太大，又要拉开档次，强化位串之间的竞争性。最常见的调整方法是线性调整法。

### 10.1.2 遗传算子

遗传算法的执行过程中，每一代有许多不同的染色体(个体)同时存在，这些染色体中哪个保留(生存)、哪个淘汰(死亡)是根据它们对环境的适应能力决定的，适应性强的有更多的机会保留下来。适应性强弱是通过计算个体适应值函数 $f(x)$的值来判别的，这个值称为适应值(fitness)。适应值函数 $f(x)$的构成与目标函数有密切关系，往往是目标函数的变种。主要的遗传算子有如下几种。

#### 1. 选择(Selection)算子

它又称复制(reproduction)、繁殖算子。

选择是从种群中选择生命力强的染色体产生新种群的过程。依据每个染色体的适应值大小来确定，适应值越大，被选中的概率就越大，其子孙在下一代产生的个数就越多。

选择操作是建立在群体中个体的适应值评价基础上的，目前常用的选择算子有以下

几种。

(1) 适应值比例法

适应值比例法是目前遗传算法中最常用的选择方法。它也叫赌轮或蒙特卡罗(Monte Carlo)选择。在该方法中,各个个体的选择概率和其适应值成比例。

设群体大小为 $n$,其中个体 $i$ 的适应值为 $f_i$,则 $i$ 被选择的概率 $P_i$ 为

$$P_i = f_i \Big/ \sum_{j=1}^{M} f_j \tag{10.1}$$

显然,概率 $P_i$ 反映了个体 $i$ 的适应值在整个群体的个体适应值的总和中所占的比例。个体适应值越大,其被选择的概率就越高。按式(10.1)计算出群体中各个个体的选择概率后,就可以决定哪些个体被选出。

(2) 最佳个体保存法

该方法的思想是把群体中适应度最高的个体不进行配对交叉而直接复制到下一代中。此种选择操作又称复制(copy)。

设在第 $t$ 代中,群体中 $a^*(t)$ 为最佳个体。而在 $A(t+1)$ 新一代群体中不存在 $a^*(t)$,则把 $a^*(t)$ 作为 $A(t+1)$ 中的第 $n+1$ 个个体(其中 $n$ 为群体大小)。

采用此选择方法的优点是,进化过程中某一代的最优解可不被交叉和变异操作破坏。但是,会使进化有可能限于局部解,即它更适合单峰性质的空间搜索。一般它都与其他选择方法结合使用。

(3) 期望值方法

① 计算群体中每个个体在下一代生存的期望数目:

$$M = f_i / \bar{f} = f_i \Big/ \sum f_i / n \tag{10.2}$$

② 若某个体被选中并要参与配对和交叉,则它在下一代中的生存的期望数目减去 0.5;若不参与配对和交叉,则该个体的生存期望数目减去 1。

③ 在②的两种情况中,若一个个体的期望值小于零,则该个体不参与选择。

对比实验表明,采用期望值法的性能高于前两种方法的性能。

(4) 排序选择方法

所谓排序选择方法是指在计算每个个体的适应值后,根据适应值大小顺序对群体中个体排序,然后把事先设计好的概率表按序分配给个体,作为各自的选择概率。所有个体按适应值大小排序,而选择概率和适应值无直接关系而仅与序号有关。这种方法的不足之处在于选择概率和序号的关系必须事先确定。此外,它和适应值比例法一样,都是一种基于概率的选择。

(5) 比例排序法

将比例法和排序法结合起来的比例排序法,即当群体中某个染色体的适应值远远大于其他染色体的适应值或群体中每个染色体的适应值相似时,按排序法进行后代选择,而在一般情形下采用比例法进行后代选择。这样既能利用两种方法各自的优点,又弥补了两种方法各自的缺点。

### 2. 交叉(Crossover)算子

它又称重组(recombination)、配对(breeding)算子。

当许多染色体相同或者后代的染色体与上一代没有多大差别时，可通过染色体重组来产生新一代染色体。染色体重组是分两步进行的，首先在新复制的群体中随机选取两个个体，然后，沿着这两个个体（字符串）随机地取一个位置，二者互换从该位置起的末尾部分。例如，有两个用二进制编码的个体 $A$ 和 $B$。长度 $L=5$，$A=a_1a_2a_3a_4a_5$，$B=b_1b_2b_3b_4b_5$ 随机选择一整数 $k\in[1,L-1]$，设 $k=4$，经交叉后变为

$$A=a_1a_2a_3|a_4a_5 \quad A'=a_1a_2a_3b_4b_5$$

$$B=b_1b_2b_3|b_4b_5 \quad B'=b_1b_2b_3a_4a_5$$

遗传算法的有效性主要来自选择和交叉操作，尤其是交叉，在遗传算法中起着核心作用。

目前有如下几种基本交叉方法。

（1）一点交叉

一点交叉又叫简单交叉。具体操作是：在个体串中随机设定一个交叉点。实行交叉时，该点前或后的两个个体的部分结构进行互换，并生成两个新个体（如上例）。

（2）二点交叉

二点交叉的操作与一点交叉类似，只是设置两个交叉点（依然是随机设定）。一个二点交叉的例子表示如下：

| | | | |
|---|---|---|---|
| | 个体 $A$ | 10 ⋮ 110 ⋮ 11 ⟶ 1001011 | 新个体 $A'$ |
| 配对个体 | 个体 $B$ | 00 ⋮ 010 ⋮ 00 ⟶ 0011000 | 新个体 $B'$ |
| | | 交叉点 1　交叉点 2 | |

由此可见，2 个交叉点分别设定在第二个基因位和第三个基因位之间以及第五个基因位和第六基因位之间。$A$、$B$ 两个体在这两个交叉点之间的码串相互交换，分别生成新个体 $A'$ 和 $B'$。对于二点交叉而言，若染色体长为 $n$，则可能有 $(n-2)(n-3)$ 种交叉点的设置。

（3）多点交叉

多点交叉是前述两种交叉的推广，有时又被称为广义交叉（generalized crossover）。

一般来讲，多点交叉较少采用，因为它会影响遗传算法的性能，即多点交叉不能有效地保存重要的模式。

（4）一致交叉

所谓一致交叉是指通过设定屏蔽字（mask）来决定新个体的基因继承两个旧个体中哪个个体的对应基因。一致交叉的操作过程表示如下：当屏蔽字位为 0 时，新个体 $A'$ 继承旧个体 $A$ 中对应的基因，当屏蔽字位为 1 时，新个体 $A'$ 继承旧个体 $B$ 中对应的基因，由此生成一个完整的新个体 $A'$。反之，可生成新个体 $B'$。显然，一致交叉包括在多点交叉范围内。一个一致交叉的例子表示如下：

| | |
|---|---|
| 旧个体 $A$ | 001111 |
| 旧个体 $B$ | 111100 |
| 屏 蔽 字 | 010101 |
| 新个体 $A'$ | 011110 |
| 新个体 $B'$ | 101101 |

### 3. 变异(Mutation)算子

选择和交叉算子基本上完成了遗传算法的大部分搜索功能,而变异则增加了遗传算法找到接近最优解的能力。变异就是以很小的概率,随机地改变字符串某个位置上的值。变异操作是按位(bit)进行的,即把某一位的内容进行变异。在二进制编码中,就是将某位 0 变成 1,1 变成 0。变异发生的概率即变异概率 $P_m$ 都取得很小(一般在 0.001~0.02 之间),它本身是一种随机搜索,然而与选择、交叉算子结合在一起,就能避免由于复制和交叉算子而引起的某些信息的永久性丢失,保证了遗传算法的有效性。

遗传算法引入变异的目的有两个:一是使遗传算法具有局部的随机搜索能力。当遗传算法通过交叉算子已接近最优解邻域时,利用变异算子的这种局部随机搜索能力就可以加速向最优解收敛。显然,此种情况下的变异概率应取较小值,否则接近最优解的模式会因变异而遭到破坏。二是使遗传算法可维持群体多样性,以防止出现未成熟收敛现象。此时变异概率应取较大值。

(1) 基本变异算子

基本变异算子是指对群体中的个体码串随机挑选一个或多个基因位并对这些基因位的基因值作变动(以变异概率 $P_m$ 做变动)。{0,1}二值码串中的基本变异操作如下:

个体 $A$ 1 0 1 1 0 1 1 —变异→ 1 1 1 0 0 1 1 个体 $A'$

变异基因位

(2) 逆转算子

逆转算子是变异算子的一种特殊形式。它的基本操作内容是:在个体码串中随机挑选两个逆转点,然后将两个逆转点间的基因值以逆转概率 $P_i$ 逆向排序。{0,1}二值码串的逆转操作如下:

个体 $A$ 10 1101000 —逆转→ 10 01011 00 个体 $A'$

逆转点

由此可见,通过逆转操作,个体中从基因位 3 至基因位 7 之间的基因排列得到逆转,即从 11010 序列变成了 01011 序列。这一逆转操作可以等效为一种变异操作,但是逆转操作的真正目的并不是变异(否则仅用变异操作就行了)而是实现一种重新排序操作。所谓重新排序是指对个体中基因排列所进行重新组合,但并不影响该个体的特征。在自然界生物的基因重组中就有这种重新排序的机制。对遗传算法而言,采用这种重新排序,目的是提高积木块(高适应度个体)的繁殖率。实际上,在用遗传算法求解某些问题时,群体中的有些个体的基因排序常常会出现这样的情况,即对形成积木块有用的某些基因分离较远,此时采用一般的交叉会破坏相应的积木块的生成。因此,有必要对这些基因进行重新排序但又不损整个个体的特征(即适应值)。

(3) 自适应变异算子

该算子与基本变异算子的操作内容类似,唯一不同的是变异概率 $P_m$ 不是固定不变的,而是随群体中个体的多样性程度而自适应调整。一般是根据交叉所得两个新个体的海明距离进行变化。海明距离越小,$P_m$ 越大,反之 $P_m$ 越小。

遗传算法中，交叉算子因其全局搜索能力而作为主要算子，变异算子因其局部搜索能力而作为辅助算子。遗传算法通过交叉和变异这一对相互配合又相互竞争的操作而使其具备兼顾全局和局部的均衡搜索能力。所谓相互配合，是指当群体在进化中陷于搜索空间中某个超平面而仅靠交叉不能摆脱时，通过变异操作可有助于这种摆脱。所谓相互竞争，是指当通过交叉已形成所期望的模式时，变异操作有可能破坏这些模式。因此，如何有效地配合使用交叉和变异操作，是目前一个重要的研究内容。

### 10.1.3 遗传算法简例

问题：求解 $f(x)=x^2$ 在[0,31]上的最大值。

**1. 初始种群**

(1) 编码：用五位二进制表示 $x$，有

$$X=0\rightarrow 0\,0\,0\,0\,0 \quad x=31\rightarrow 1\,1\,1\,1\,1$$

(2) 初始种群

随机产生 4 个个体：13，24，8，19(分别用二进制表示)

(3) 适应值 $f_i$

直接用目标函数作为适应值：$f(x)=x^2$

① 非负；　② 逐步增大

(4) 选择率 $p_s$ 和期望值

选择率：$p_{si}=f_i/\sum f_i$

平均适应值：$\underline{f}=\sum f_i/n$

期望值：$f_i/\underline{f}$

(5) 实选值

期望值取整数，具体计算如表 10.1 所示。

表 10.1　初始种群参数计算

| 编号 | 初始种群位串 | 参数值 $x$ 值 | 目标适应值 $f(x)=x^2$ | 选择率 $f_i/\sum f_i$ | 期望值 $f_i/\underline{f}$ | 实选值 |
|---|---|---|---|---|---|---|
| 1 | 0 1 1 0 1 | 13 | 169 | 0.14 | 0.58 | 1 |
| 2 | 1 1 0 0 0 | 24 | 576 | 0.49 | 1.97 | 2 |
| 3 | 0 1 0 0 0 | 8 | 64 | 0.06 | 0.22 | 0 |
| 4 | 1 0 0 1 1 | 19 | 361 | 0.31 | 1.23 | 1 |
| 总和 $\sum$ | | | 1170 | 1.00 | 4.00 | 4.0 |
| 平均值 | | | 293 | 0.25 | 1.00 | 1.0 |
| 最大值 | | | 576 | 0.49 | 1.97 | 2.0 |

### 2. 遗传第一代

参数计算如表 10.2 所示。

**表 10.2　初始种群遗传过程**

| 选择后的交配池（下划线部分交叉） | 交叉对象（随机选择） | 交叉位置（随机选择） | 新的种群 | $x$ | $f(x)=x^2$ |
|---|---|---|---|---|---|
| 0　1　1　0　$\underline{1}$ | 2 | 4 | 0　1　1　0　0 | 12 | 144 |
| 1　1　0　0　$\underline{0}$ | 1 | 4 | 1　1　0　0　1 | 25 | 625 |
| 1　1　$\underline{0\ \ 0\ \ 0}$ | 4 | 2 | 1　1　0　1　1 | 27 | 729 |
| 1　0　$\underline{0\ \ 1\ \ 1}$ | 3 | 2 | 1　0　0　0　0 | 16 | 256 |
| 总和 $\sum$ | | | | | 1754 |
| 平均值 | | | | | 439 |
| 最大值 | | | | | 729 |

具体说明如下：

（1）选择（繁殖）

在种群中，实选值（期望值）高者多繁殖；实选值（期望值）低者少繁殖或不繁殖。繁殖（复制）的个体放入交配池中。

（2）交叉

随机选择交配对象（相同个体不交配），如个体 1 和 2，3 和 4。随机选择交叉点进行交叉。

（3）变异

取变异概率 $P_e=0.01$，表示每 100 个体中有一个个体的一位发生变异。上例中未进行个体变异。

遗传得到的新的种群，其平均值和最大值都有很大提高。

均值：293→439

最大值：576→729

新种群中四个个体，有 2 个变好：25，25；有 2 个变坏：12，16。

### 3. 遗传第二代

新种群的参数计算如表 10.3 所示，新种群的遗传过程如表 10.4 所示。

**表 10.3　新种群参数计算**

| 编号 | 初始种群位串 | 参数值 $x$ 值 | 目标适应值 $f(x)=x^2$ | 选择率 $f_i/\sum f_i$ | 期望值 $f_i/\bar{f}$ | 实选值 |
|---|---|---|---|---|---|---|
| 1 | 0 1 1 0 0 | 12 | 144 | 0.08 | 0.33 | 0 |
| 2 | 1 1 0 0 1 | 25 | 625 | 0.36 | 1.42 | 1 |
| 3 | 1 1 0 1 1 | 27 | 729 | 0.42 | 1.66 | 2 |

续表

| 编号 | 初始种群位串 | 参数值 $x$ 值 | 目标适应值 $f(x)=x^2$ | 选择率 $f_i/\sum f_i$ | 期望值 $f_i/\bar{f}$ | 实选值 |
|---|---|---|---|---|---|---|
| 4 | 1 0 0 0 0 | 16 | 256 | 0.15 | 0.58 | 1 |
| 总和 $\sum$ | | | 1754 | 1.00 | 4.00 | 4.0 |
| 平均值 | | | 439 | 0.25 | 1.00 | 1.0 |
| 最大值 | | | 729 | 0.42 | 1.66 | 2.0 |

**表 10.4　新种群的遗传过程**

| 选择后的交配池（下划线部分交叉） | 交叉对象（随机选择） | 交叉位置（随机选择） | 新的种群 | $x$ | $f(x)=x^2$ |
|---|---|---|---|---|---|
| 1 <u>1 0 0 1</u> | 2 | 1 | 1 1 0 1 1 | 27 | 729 |
| 1 <u>1 0 1 1</u> | 1 | 1 | 1 1 0 0 1 | 25 | 625 |
| 1 1 0 <u>1 1</u> | 4 | 3 | 1 1 0 0 0 | 24 | 576 |
| 1 0 0 <u>0 0</u> | 3 | 3 | 1 0 0 1 1 | 19 | 361 |
| 总和 $\sum$ | | | | | 2291 |
| 平均值 | | | | | 572 |
| 最大值 | | | | | 729 |

单纯用交叉而没有用变异，则遗传多少代是得不到最优解 31(11111)。主要是第三位所有个体都是 0，这样只能得到 27(11011) 次优解。

若在第四位中挑选一个个体进行变异，由 0 变成 1，再进行遗传将会得到最优解。

### 10.1.4　遗传算法的特点

遗传算法是模拟自然选择和生物遗传机制的优化算法，利用三个遗传算子产生后代，通过群体的迭代，使个体的适应性不断提高，最终群体中适应值最高的个体即是优化问题的最优或次优解。遗传算法与传统的优化方法有不同的特点。

#### 1. 遗传算法是进行群体的搜索

传统的优化方法是从一个点开始搜索。例如爬山法(Climbing)是从当前点邻近的点中选出新点，如果新点的目标函数值更好，那么该新点就变成当前点，否则就选择和测试其他邻近点。如果目标函数值没有更进一步地改进，则算法终止。很显然，爬山法只能提供局部最优解，它依赖于初始点的选择。

遗传算法是对多个个体进行群体的搜索，即在问题空间中不同区域进行搜索，构成一个不断进化的群体序列。对于复杂问题的多峰情况，遗传算法也能以很大的概率找到全局最优解。

**2. 遗传算法是一种随机搜索方法**

遗传算法使用三个遗传算子,选择算子通过选择概率复制个体。交叉算子通过交叉概率在交配池中决定配对的个体是否需要进行交叉操作。变异算子通过变异概率确定某些基因位上值进行变异。可见,三个遗传算子都是随机操作,能产生好的后代,引导其搜索过程朝着更优化的解空间移动。可见遗传算法虽然是一个随机搜索方法,但它是高效有方向的搜索,而不是一般随机搜索方法那种无方向的搜索。

**3. 遗传算法处理的对象是个体,而不是参变量自身**

遗传算法要求将优化问题的参变量编码成长度有限的位串个体,即参变量是个体的组成部分。通过遗传算子的随机变换操作位串个体,并从中找出高适应值的位串个体。遗传算法不是对参数变量进行直接操作。

编码操作可直接对结构对象进行操作。结构对象泛指集合、序列、矩阵、树、图、链和表等一维或二维结构形式的对象。这一特点使得遗传算法具有广泛的应用领域。

**4. 遗传算法不需要导数或其他辅助信息**

一般传统的搜索算法需要一些辅助信息,如梯度算法需要求导数,当这些信息不存在时(如函数不连续时),这些算法就失效。而遗传算法只需要适应值信息,用它来评估个体,引导搜索过程朝着搜索空间的更优化的解区域移动。

**5. 隐含并行性**

遗传算法实质上是模式的运算。对于一个长度为 $l$ 的串,其中隐含着 $2^l$ 个模式。若群体规模为 $n$,则其中隐含的模式个数介于 $2^l$ 和 $n \cdot 2^l$ 之间。Holland 指出,遗传算法实际上是对 $n$ 个位串个体进行运算,但却隐含地处理了大量的模式,这一性质称为隐含并行性(implicit parallelism)。

隐含的并行性是遗传算法优于传统的搜索方法的关键所在。

## 10.2 基于遗传算法的分类学习系统

### 10.2.1 概述

1978 年 Holland 等实现了第一个基于遗传算法的机器学习系统 CS-1。该系统由消息表(message list)、分类器(classifier)的字符串规则、遗传算法及一个信息分配机制组成。他还提出了桶队(bueket brigade)算法。1980 年 Smith 实现了分类器系统 LS-1。尽管 LS-1 诞生于 CS-1 之后,但 LS-1 系统在若干重要的方面与 CS-1 有根本性的差别。具体表现在字符串规则、染色体表示方法、搜索结构的形成以及遗传操作算子的应用上。LS-1 系统影响更大。

分类器系统是一种对字符串规则(又称分类器)的学习系统,它由规则与消息(rule and message)系统、信任分配(apportionment of credit)系统及遗传算法三个主要部分组成,其中规

则与消息系统是产生式系统的一种特殊形式。产生式规则的一般形式为：IF<condition> THEN<action>。它具有计算完备性，且其描述也较方便，一条规则或一个规则集往往能将一种复杂的情况非常紧凑地描述出来。因而它为众多的专家系统所采用。在分类器系统中，对产生式规则的语法做了很大的限制，采用了定长的表示形式，从而适于采用遗传操作。

传统的专家系统在每一次匹配中采用单条规则激活的串行运行方式。分类器系统采用了并行激活方式，即在每一匹配周期，它允许多条规则被同时激活，只有在出现两个互斥的动作或当匹配的规则集大小超出消息表的容量时，才考虑规则的选择问题。

传统的专家系统中的规则和规则相应的重要程度(Strength)是事先由程序设计者根据专家经验给出的，是固定不变的。而分类器系统是一个自适应的学习系统，获取的规则和相应的重要程度是不固定的。

### 10.2.2 遗传分类学习系统 GCLS 的基本原理

我们研制了一种新的遗传分类器学习系统(Genetic Classifier Learning System, GCLS)，与基本的分类器系统相比，GCLS 系统采用了训练和测试同时进行的策略，使得系统能够在训练后继续学习，从而能更好地适应不断变化的客观环境。GCLS 系统还设计了工作和精练两种不同的分类器，通过精练分类器中对规则的进一步处理，减少了所获规则的冗余性。GCLS 系统中设计的信任分配机制可有效地处理训练样本带有噪声和异常特例等问题，同时体现了规则与训练样本的统计规律，使得判别结果容易用背景知识进行定性、定量相结合的解释，从而可获得与客观环境相容的判别规则。

#### 1. GCLS 系统结构

遗传分类学习系统 GCLS 的结构如图 10.2 所示。

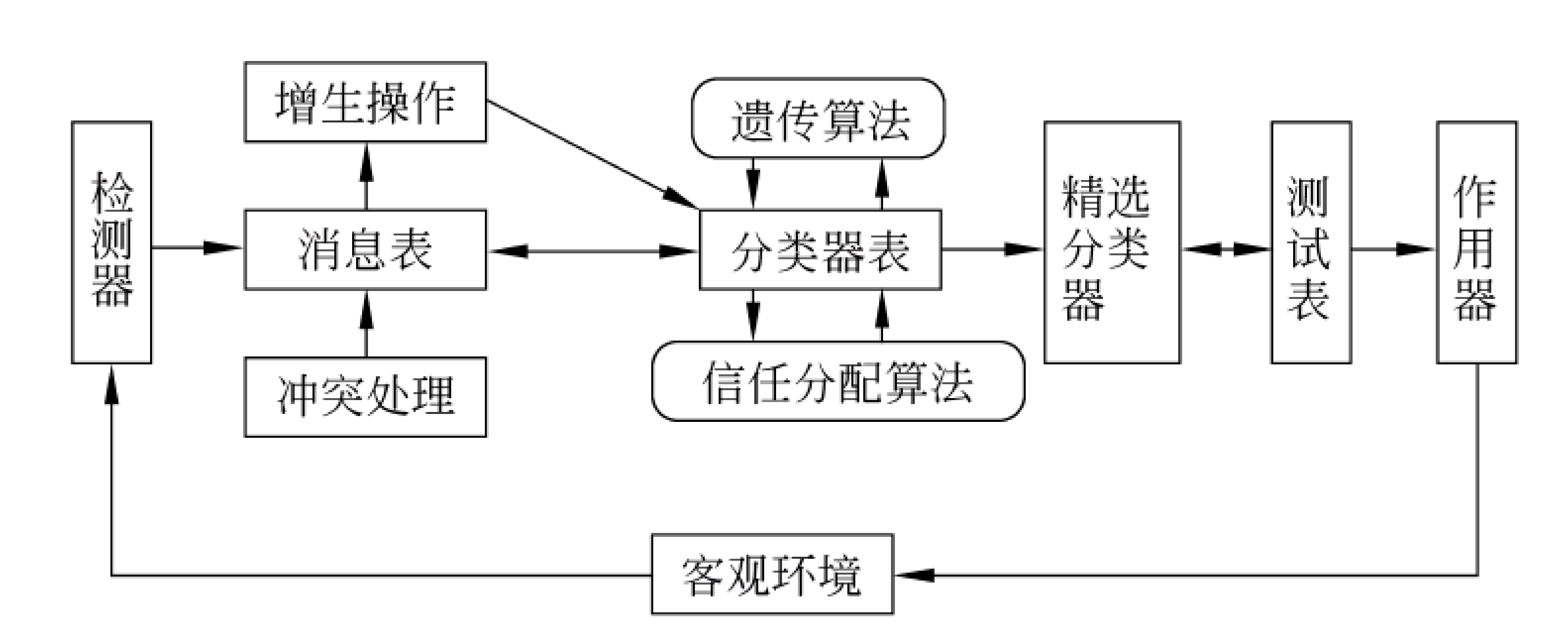



图 10.2 遗传分类学习系统 GCLS 的结构

客观环境信息通过分类器系统的检测器(Detector)被编码成有限长的消息(Messages)。然后发往消息表；消息表中的消息触发位串规则(称为分类器)，被触发的分类器又向消息表发消息，这些消息又有可能触发其他的分类器或引发一个行动，通过作用器(Effector)作用于客观环境。

(1) 检测器

检测器(Detector)将环境信息由条件部分和结论部分组成的训练的例子集编码成二进制字符串的消息。一条消息 $M_i$ 是一个二元组，其形式如下：$M_i=[x_i, y_i]$

其中：$i$ 为消息号；$x$ 为条件部分，即训练例子的各特征编码，$x_i \in \{0,1\}^n$。$y$ 为结论部分，即训练例子的类别，$y_i \in \{0,1\}^m$。例如：[(10001011)，(1011)]是一条由一个 8 位条件和 4 位结论组成的消息。

(2) 消息表

消息表(Message List)包含当前所有的消息(训练例子集)。每个消息由条件部分(Condition)和结论部分(Action)组成。

(3) 分类器

分类器(Classifier)系统与一般的机器学习系统不同，它最后所获得的规则中包含通配符＃，这就会出现大量的冗余规则，如 1＃＃0，1110 是一致的。一般来说，应该使系统产生最小的规则集获得较高的性能。规则集越小，系统的时间性能就越好。

一个分类器是由当前遗传产生的一条规则组成，分类器表由所有分类器组成，构成了规则集。一个规则 $C_i$ 是一个三元组，形式如下：

$$C_i = [U_i, V_i, \text{fitness}_i]$$

其中，$U_i$ 是条件部分，$U_i \in \{0,1,\#\}^n$，＃表示通配符；$V_i$ 是结论部分(action)，$V_i \in \{0,1\}^m$；$\text{fitness}_i$ 是规则 $i$ 的适应值，它又是一个二元组，其形式如下：

$$\text{fitness}_i = [\text{fit1}, \text{fit2}]$$

其中：fit1、fit2 均为正整数，分别表示在该规则覆盖的范围内，与规则结论一致和不一致的消息个数。

在分类器中，将最后获得的规则放入精练分类器中。

(4) 测试表

测试表(Test List)是由所有测试例子组成，一个测试例子 $T_i$ 也是一个同消息形式一样的二元组，只是它的结论部分 $y_i \in \{*\}^m$，* 表示未确定。当它到精练分类器匹配规则后，其结论部分 $y_i$ 就被赋值成与消息 $M_i$ 完全一样的形式，即 $y_i \in \{0,1\}^m$，变成一条新的消息。结论可直接作用于环境，也可通过环境将新消息反馈给系统，以便系统能继续学习下去，从而更好地适应不断变化的客观环境。

(5) 作用器

作用器(Effector)将所有测试例子的判别结果(类别)转换成具体问题的输出值，并作用于环境。

**2. GCLS 系统的主要算法**

(1) 信任分配算法

信任分配算法(Credit Assignment Algorithm，CAA)实质是对分类器表中各条规则作用于环境的有效性进行评价，而本系统中的环境就是前面所说的训练例子集，将分类器表中的规则与消息表中的消息逐个匹配，根据匹配的成功与否，来修改规则的适应值，以保证好的规则的生存，以及不适应的规则的消亡，其主要步骤如下：

① 初始化规则的适应值，即 fit1←0，fit2←0。

② 从消息表[M]中取出一条消息，与分类器表中的规则逐个进行比较。

IF　条件(condition)和结论(action)均匹配,THEN　fit1=fit1+1;

IF　条件匹配,结论不匹配,THEN　fit2=fit2+1;

IF　条件不匹配,THEN　fitness← fitness

③ 返回步骤②,直到[M]中的消息全部取完。

(2) 遗传算法

遗传算法(Genetic Algorithms)是用来产生新的规则。在GCLS系统中,遗传算法的调用是在分类器表中每一新的种群产生之后,系统采用了一种限制交配策略,也就是本地算子中的受限交配,即只允许同类(规则的结论部分相同)的规则进行交叉。这样,对同一结论的规则,只允许其条件部分进化。假如规则的条件和结论同时进化,就可能引起种群不收敛的情况产生。此外,产生的新规则并不取代老规则,而是与老规则合并到一起,形成工作分类器的新的初始种群。

GCLS中遗传算法的主要步骤如下:

① 在分类器表中,根据与各规则适应值成正比的概率,选择复制出 $K$ 个规则。

本系统中采用了比例法来选择复制。按 $f_i/\sum f_j$ 取整$\left(f_i\right.$ 是 $X_i$ 的适应值;$\sum f_j$ 是种群中各规则的适应值之和$\left.\right)$,来决定第 $i$ 个规则在下一代中应复制其自身的数目 $k_i$,而 $K=\sum k_i$。

② 采用遗传算子(交叉、变异),重新产生 $k$ 个新的规则。

在GCLS中,按一定的概率 $P_c$ 从①中随机选择出一对规则进行交叉,同样,也是按一定的概率 $P_m$ 对规则中的某些位进行变异。这里的交叉概率 $P_c$ 和变异概率 $P_m$ 都是经验参数,在不同应用问题中的取值都是不同的。

(3) 合并操作

采用合并操作(Merge Operation)旨在减少冗余规则。

① 对于分类器表中初始种群的每一规则,若其对应的fit1恒不等于0,且fit2等于0,则保留,否则淘汰。

② 将保留下来的规则两两匹配。设R1、R2为两个保留下来的规则。

IF R1⊇R2,且fit1(R1)=fit1(R2),THEN 保留R2,淘汰R1。

IF R1⊇R2,且fit1(R1)>fit1(R2),THEN 保留R1,淘汰R2。

(4) 冲突处理

一般的分类器系统不包括矛盾例子的处理,而在实际应用领域尤其在预测领域,这种情况经常出现,如天气预报。所以系统要能够对这些矛盾例子进行处理。GCLS系统中设计的冲突处理(Conflict Process)是将消息表[M]中的消息两两匹配,对于那种只有条件匹配,而结论不匹配的消息作为冲突消息记录下来,并都从[M]中删除,即在分类器中删除已生成的冲突规则。

(5) 增生操作

如果分类器表中没有一个与消息匹配的规则,则用增生操作(Supplement Operation)生成一个与之相匹配的规则。在消息位串上对条件部分的每一位按系统给定的#的生成率进行变异。若发生变异则由1或0改为#,否则不变。然后将变异过的消息作为新的规则

的条件部分，结论部分保留消息中的结论。新生成的规则加入到分类器表中的方法有两种：一是用新生成的规则置换掉分类器表中的适应值最小的规则；二是直接加入到分类器表中，只有当分类器表的增长超过一定限度时才进行淘汰。这样做的好处是在系统运行的初期，当适应值的强弱差别还不明显时，能较好地避免将有发展潜力、好的规则淘汰掉。在本系统中，采用了后一种方法。

此外，在 GCLS 系统中采用了训练与测试同时进行。一般的分类器系统同现存的机器学习系统一样：训练与测试是分开进行的，规则的获取完全依赖于训练例子的选取的好坏。例如，训练例子中正反例的比例应与实际问题中正反例的比例相同，这一般是不可能做到的，且选取的训练例子不可能包含实际问题中的所有情况。而 GCLS 的这种策略使系统能在训练后继续学习，这就能保证不依赖于选取的例子。从而能更好地适应不断变化的客观环境，得到更符合实际的规则。

### 3. GCLS 系统获取规则的过程

遗传分类学习系统 GCLS 系统的学习过程就是一个获取规则的过程。GCLS 规则生成流程如图 10.3 所示。

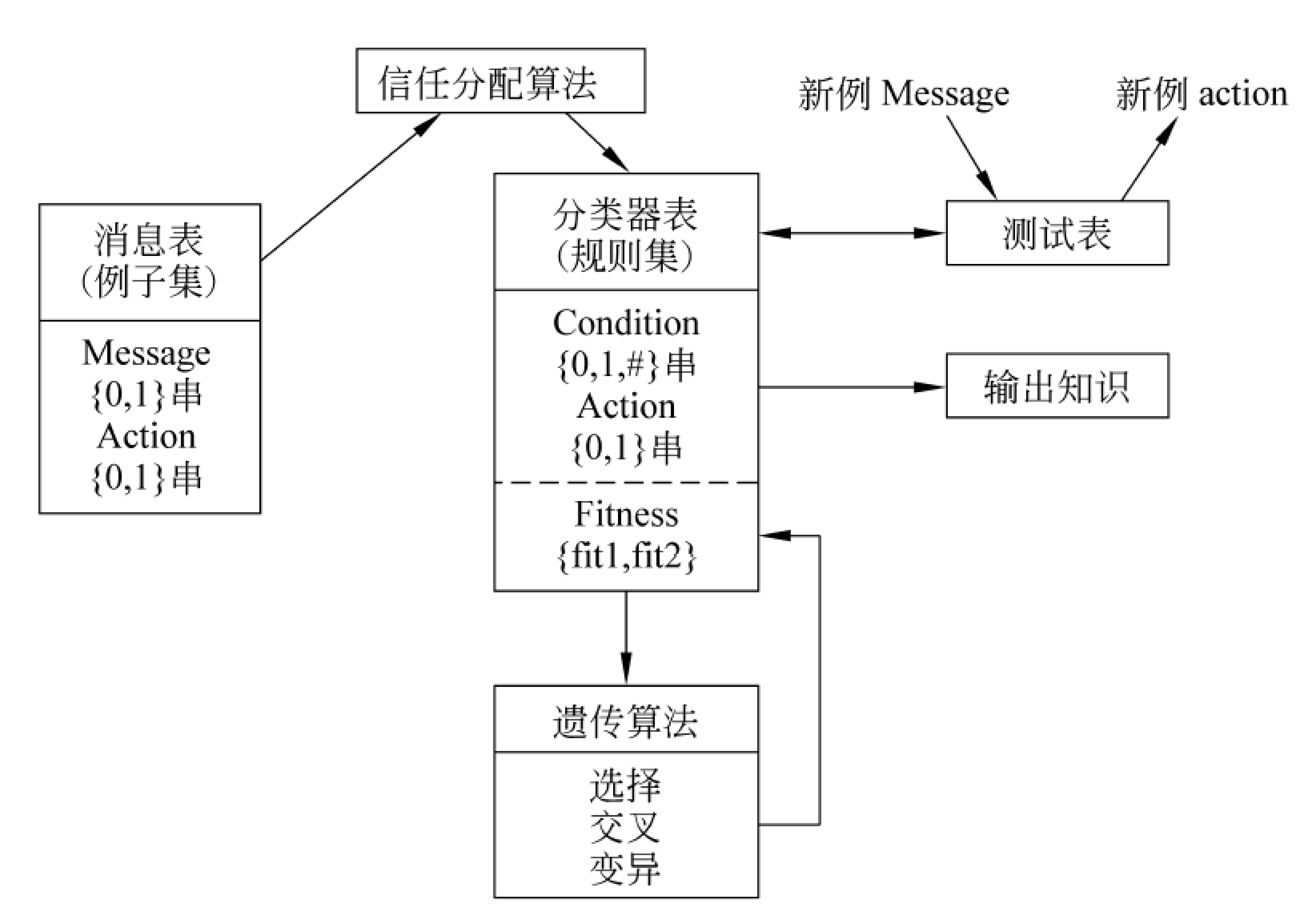



图 10.3　GCLS 规则生成过程

规则的获取是通过初始化一个随机的种群(分类器)，而后触发系统的信任分配机制和遗传算法等操作，直到获得一组源于环境信息(训练集)的、达到期望状态或特征的规则(分类器)，再把最后获得的规则拷贝到一个精炼分类器中，以供下一步测试未知例子的类别使用，至此，GCLS 系统的一个学习过程就已结束。

在 GCLS 系统中一次学习过程的结束是当前分类器已收敛，即种群的规则与其父代完全相同，并且各规则的适应值已连续 $p$ 次保持不变，也就是说当前工作种群已不再进化了，$p$ 是系统根据不同的应用问题而事先设置的一个参数，在本系统应用实例中 $p$ 均取 100。

GCLS 系统的执行步骤可概括如下：

(1) 初始化 GCLS 的所有预置参数。(如分类器表中初始规则数目 $n$；交叉、变异概率

$P_c$、$P_m$；判断分类器收敛的参数 $p$ 等)；初始化分类器表，设为初始种群 0，随机产生 $n$ 个规则，并给每个规则赋一个相等的初始适应值。

(2) 将环境信息(训练集)通过检测器编码成二进制消息放入消息表[M]中。

(3) 对[M] 进行冲突处理。将[M]中的消息进行两两匹配，把只有条件匹配而结论不匹配的消息做冲突处理后，直接送往精炼分类器中。

(4) 对初始种群 0 调用信任分配算法，修改其中的规则适应值。如果种群 0 中无一与消息匹配的规则，则进行增生操作，生成一个相匹配的规则，将该规则直接加入到种群 0 中。

(5) 对种群 0 进行合并操作，合并后的种群设为种群 1。

(6) 假如种群 1 已收敛，则拷贝该种群的规则到精炼分类器中，转向步骤(9)。

(7) 调用遗传算法，生成新一代种群 2，将其与种群 1 合并，而后送给种群 0，从而形成新的种群 0。

(8) 返回步骤(4)。

(9) 对测试表[T]调用精炼分类器规则，生成[T]的结论部分。

(10) 将[T]送往作用器，转换成实际的输出值以作用于环境。

### 10.2.3 遗传分类学习系统 GCLS 的应用

#### 1. 应用说明

这是一个学习识别脑出血和脑血栓两种疾病的诊断规则的应用实例，这个问题实际上是从大量已知患者病例(训练例子集)中找到这两类病的识别规则。

在这一应用实例中，实际上只有两种类别：A 脑出血；B 脑血栓。

为了做出判断，应当考虑如下几个方面的特征(属性)：

(1) 病人的既往史，包括 a 高血压(有 01，无 00)；b 动脉硬化(有 01，无 00)；

(2) 起病方式(快 01，慢 00)；

(3) 局部症状，包括：

a. 偏瘫(是 01，否 00)；

b. 瞳孔不等大(是 01，否 00)；

c. 两便失禁(是 01，否 00)；

d. 语言障碍(是 01，否 00)；

f. 意识障碍(无 00，深度 01，轻度 10)；

(4) 病理反射(阳 01，阴 00)；

(5) 膝腱反射(无 00，活跃 01，不活跃 10)；

(6) 病情发展(快 01，慢 00)。

上面是从 6 个方面 12 个特征来识别诊断患者到底得的是脑出血还是脑血栓。

#### 2. 获取知识

从 60 个脑出血和脑血栓病人的病例中选出 30 个病例作为训练样本，30 个作为测试样本。

本实例采用二进制编码方式。每个训练例子都是由 12 个特征和 1 个类别组成的，每个特征和类别都由 2 位二进制字符表示的。那么，将例子编码成二进制字符串的消息就是一个由 24 位条件和 2 位结论组成的二元组，例如消息 M＝[(010001010101010110100101)，(01)]。

训练集是由 15 个脑出血和 15 个脑血栓患者组成 30 个训练样本。本实验在对 30 个训练样本进行学习后，得到 12 个规则：学习终止于第 170 代。

获取的主要规则如下：

(1) 高血压＝有∧瞳孔不等大＝是∧膝腱反射＝不活跃 →脑出血(11)
(2) 瞳孔不等大＝是∧语言障碍＝是 →脑出血(12)
(3) 高血压＝有∧起病方式＝快∧意识障碍＝深度 →脑出血(13)
(4) 高血压＝有∧病情发展＝快 →脑出血(15)
(5) 高血压＝有∧动脉硬化＝有∧起病方式＝慢 →脑血栓(13)
(6) 动脉硬化＝有∧病情发展＝慢 →脑血栓(15)
(7) 动脉硬化＝有∧意识障碍＝无 →脑血栓(12)

以上括号内的数值表示该规则的适应值。

## 10.3 进化计算

### 10.3.1 进化计算概述

进化计算(Evolutionary Computation，EC)是模拟自然界生物进化过程中群体随机搜索技术和自然选择法则，即通过进化过程完成问题的求解。

进化计算最典型的方法有 4 种：遗传算法(Genetic Algorithm，GA)、进化策略(Evolutionary Strategy，ES)、进化规划(Evolutionary Programming，EP)和遗传程序设计(Genetic Programming，GP)。历史上这 4 种算法是彼此独立发展起来的。这些方法虽然在基因结构表达方式及对交换与突变作用的侧重点上有所差异，但它们的原理都是借鉴生物界中进化与遗传机理来解决复杂的工程技术问题。

进化计算起源于 20 世纪 30 年代通过仿真生物进化过程进行机器学习的研究。在 1932 年，Cannon 就把自然进化想象为一个学习过程，与自然进化过程的机制和结果稍微不同的是，Cannon 不是通过维持一个特定的种群来进行搜索，而是对单个个体反复进行随机试验。1959 年 Friedman 推测，利用变异和选择的仿真可以设计“思想机器”，并且指出下棋的程序可以用这种方法设计。在 1960 年，Cambell 猜想：在导致知识扩张的所有过程中，都要涉及“盲目—变化—选择—幸存”的过程。

遗传算法 GA 是由美国的 J. Holland 于 1975 年在前人的基础上创建的，后由 K. DeJong、J. Grefenstette、D. Goldberg 和 L. Davis 等人进行了改进。

进化规划 EP 是由美国的 L. J. Fogel、A. J. Owens 和 M. J. Walsh 于 1962 年提出的，最近又由 D. B. Fogel 进行了完善。

进化策略 ES 是由德国的 I. Rechenberg 和 H. P. Schwefel 于 1965 年提出的。

遗传程序设计 GP 是由美国的 John R. Koza 于 1992 年正式提出的。

1990 年以后，遗传算法与进化规划和进化策略开始有所交流，并接触到对方的研究工作，他们发现彼此在研究中所依赖的基本思想都是基于生物界的自然遗传和自然选择等生物进化思想，于是将这类方法统称为进化计算，相应的算法称为进化算法或进化程序。1993 年《进化计算》这一专业领域的第一份国际杂志问世；1994 年 IEEE 神经网络委员会主持召开了第一届进化计算国际会议。

群体搜索策略和群体中个体之间的信息交换是进化算法的两大特点。它们的优越性主要表现在：

(1) 进化算法在搜索过程中不容易陷入局部最优，即使在所定义的适应度函数是不连续的，非规则的或有噪声的情况下，它们也能以很大的概率找到全局最优解。

(2) 由于它们固有的并行性，因此进化算法非常适合于巨量并行机。

(3) 进化算法采用自然进化机制来表现复杂的现象，能够快速可靠地解决非常困难的问题。

(4) 由于它们容易同别的技术混合，进化算法目前已经在最优化、机器学习和并行处理等领域得到了越来越广泛的应用。

### 10.3.2 进化策略与进化规划

#### 1. 进化策略

早期的进化策略的种群中只包含一个个体，并且只使用变异操作。在每一代中，变异后的个体与其父代进行比较，并选择较好的一个，这种选择策略被称为(1+1)策略。进化策略的一般算法可以描述如下：

(1) 问题为寻找实值 $n$ 维矢量 $x$，使得函数 $F(x)$：$R^n \to R$ 取极值。不失一般性，设此程序为极小化过程。

(2) 从各维的可行范围内随机选取样本 $x_i, i=1,\cdots,p$ 的初始值。初始试验的分布一般是均匀分布。

(3) 通过对于 $x$ 的每个分量增加零均值和预先选定的标准差的高斯随机变量，从每个样本 $x_i$ 产生子代 $x_i'$。

(4) 将函数 $F(x_i)$ 和 $F(x_i')$，$i=1,\cdots,p$ 的差进行排序，选择并决定那些矢量保留。具有最小误差的 $p$ 个矢量变成下一代的新样本。

(5) 进行新试验，选择具有最小方差的新子代，直到获得充分解，或者直到满足某个终止条件。

在这个模型中，把试验解的分量看做个体的行为特性，而不是沿染色体排列的基因。假设不管发生什么遗传变换，所造成各个行为的变化均遵循零均值和某个标准差的高斯分布。由于基因多效性和多基因性，特定基因的改变可以影响许多表现型特征，所以在创造新子系时，较为合适的是同时改变亲本所有分量。

进化策略初始试验采用上述算法，主要采用单样本—单子代的搜索，即"(1+1)进化策略，((1+1)−Es)"，其中单个子代是由单个样本产生的，它们都被置于生存竞争中，较弱的

一个要被挑选出来消去。

1981 年,Schwefel 在进化策略中使用多重样本和子代,这是对 Rechenberg 早期工作(使用多重样本,但是仅使用单个子代)的发展。

**2. 进化规划**

Fogel 认为,智能行为需要有如下的复合能力:①预报它的环境;②把预报变成对于给定目标的适当响应。

进化规划中经常用有限状态自动机(FSM)表示在环境(有限字母符号序列)上运行的算法。1966 年 Fogel 等人进行了一系列实验。

例如,考虑把递增的自然数划分为质数(表示为“1”)或合数(表示为“0”)所产生的非平稳序列。这样环境由序列“01101010001…”所组成,分别代表了正整数“1,2,3,4,5,6,7,8,9,10,11,…”是质数还是合数的情况。当然存在直接测试是否为质数的方法,但是我们要做的是,依据观察到的质数与合数序列,预测下一个整数是否为质数,这就不太容易了。采用收益函数为“有无函数”,即对于每个正确的预报,收益为 1,而对于每个错误的预报,收益为 0。并且将所得的结果减去机器的状态数,乘以 0.01 进行修正,用此修正项对个体复杂性进行惩罚。

试验得出,在初始时波动大(由于采样数太少),在第 115 个符号处预报正确率增到 78%,然后基本维持不变,一直到第 200 个数。到 719 个符号以后,累积的正确预报百分比达到 81.9%。逐渐趋于 100%的正确率,这是因为质数越变越稀,机器将不断预报“合数”。

Fogel 等人在 1966 年发现,该进化算法学会了能被 2 或 3 整除的数是合数。1968 年,Fogel 的实验表明,进化规划能成功地“认识到环境中的循环性,发现能被 2 整除的数不是质数”等等。换句话说,虽然程序没有关于质数本性和除法能力的先验知识,也能够归纳出质数的定义。但是可以看出,进化规划不是一个完善的预报器,因为有限状态机无法表示产生质数的算法。

1969 年,Fogel 和 Burgin 将进化规划用于博弈(协进化)。他们进行了一系列两人零和博弈实验。在只有少量玩法(例如 4 种)的简单博弈中,进化规划都能够发现总体最优策略,在一些更复杂的环境下,进化规划产生的策略在性能上已经优于人的策略。进化规划还可以推广到非零和博弈隋况,例如追踪—逃逸问题。

进化规划的计算流程包括:

(1) 确定问题的表达方式。

(2) 随机产生初始群体,并计算其适应值。

(3) 用如下操作产生新群体:①变异,对旧个体添加随机量,产生新个体;②计算新个体适应值;③选择,挑选优良个体组成新群体。

(4) 反复执行(3),直到满足终止条件,选择最佳个体作为进化规划的最优解。

其中,进化规划的变异算子的自适应调节功能,主要依靠适应值 $f(X)$ 来实现,表达式如下:

$$x_i' = x_i + \sqrt{f(x)} \cdot N_i(0,1)$$

式中:$x_i$ 为旧个体目标变量的第 $i$ 个分量;$x_i'$ 为新个体目标变量的第 $i$ 个分量;$\sqrt{f(x)}$ 为

旧个体 $x$ 的适应值；$N_i(0,1)$为对第 $i$ 个分量发生的随机数，服从标准正态分布。

**3. 遗传程序设计**

遗传程序设计运用遗传算法的思想，常采用树的结构来表示计算机程序。1989 年，美国斯坦福大学的 Koza 基于自然选择原则创造性地提出了用层次化的计算机程序来表达问题的遗传程序设计(Genet Ic Programming，GP)方法，成功地解决了许多问题。

利用进化去搜索一个可表示计算机程序的树结构空间，使用一些遗传操作动态地改变这些结构，找到解决该问题的可行的计算机程序。这种广义的计算机程序进化的结构本身是计算机程序，能够根据环境状态自动改变程序的结构及大小，从而可以更灵活地表达复杂的事物的性质。

GP 最初由一随机产生的计算机程序群体开始，这些计算机程序由适合于问题空间领域的函数所组成，这样的函数可以是标准的算术运算函数、标准的编程操作、逻辑函数或由领域指定的函数。群体中每个计算机程序个体都是用适应值测试来评价的，该适应值与特定的问题领域无关。

GP 的操作步骤：

(1) 确定个体的表达方式，包括函数集和终止符集等。

(2) 随机生成初始群体，它由关于问题(计算机程序)的函数随机组合而成。

(3) 计算各个体的适应值，即执行每个计算机程序，根据其解决问题的能力，为其指定一个适应值。

(4) 根据遗传参数，通过复制、交换和突变产生新个体(计算机程序)：① 复制，将已有的优良个体复制，加入新群体中，并相应删除劣质个体；②交换，将选出的 2 个个体进行交换，所产生的 2 个新个体插入新群体中；③变异，随机改变个体某一部分，将新个体插入新群体中。

(5) 反复执行(3)和(4)，直至获得满意结果。此时，后代中适应值最高的计算机程序个体被指定为 GP 的结果，这一结果可能是问题的解或近似解。

GP 的主要特点在于它是可变长的、层次化的、常常是树结构的遗传材料，而且大多数情况下，程序个体是可执行的，也就是说常常通过某类解释器解释程序。

GP 方法可应用于许多领域，如电子工程、化学、财政、经济、生命科学、艺术等。

(1) 预测和分类：使用历史数据库来预测新事例，如商业化地应用到保险、气象预报、财政等领域，还可进行时间序列预测、蛋白质形状预测等。

(2) 人工生命：用计算机模拟生物的自然进化或发现规律。

(3) 神经网络设计：设计神经网络结构、发现学习规则和相关权值，以使神经网络完成指定任务。

(4) 图像和信号处理：图像识别、图像恢复、图像和声音的压缩等。

### 10.3.3 进化计算小结

**1. 交叉和变异的关系**

进化计算研究领域中主要争议的问题之一是，遗传算子中是变异还是交叉重要。进化

规划和进化策略学派都强调变异算子的重要性,并将它作为主要的遗传算子,近来的研究也证实变异算子非常有效,D. Fogel 强烈地声称:在一般意义上,交叉并不优于变异。

在另一方面,遗传算法学派坚信交叉是更有效的算子,在分析交叉和它对性能影响上做了大量的工作,这些研究几乎都认为变异是辅助算子,是次重要的。最近 Schaffer 和 Eshelman 在试验中比较了变异和交叉得出结论:仅有变异并不总是足够的。

Spears 从定义交叉和变异的两个潜在作用——分裂(Disruption)和构造(Construction)出发,考虑它们在执行这两个作用上的差异,研究结果表明:对于分裂,变异比交叉有效,虽然它缺少交叉保留个体共同等位基因的能力;然而对于构造,交叉比变异更有效。

关于变异和交叉相对重要性的问题,可以在更高的层次上来看待,变异用于群体中产生随机多样性,而交叉相当于一个加速器,由部分加速构成整体行为,从而原来的问题就转化为多样性和构造的相对重要性。对于遗传算法,这也与探测(Exploration)和开发(Exploitation)之间的权衡有关。多样性和构造的相对重要性是解答 Holland 体系和 Fogel 体系之间差异的关键。特别地,Fogel 等人怀疑交叉的重要性,他们不相信自然选择会选择个别的特性或特性的组合,交叉被看成是第三位的因素,因为在自然中它似乎不经常出现。

变异和交叉都不该轻易地提倡和舍弃;每个算子在搜索中起着不同的作用。对于一个问题事先确定哪个算子更重要很难。为得到好的性能,探测和开发之间达到适当的平衡依赖于群体中多样性的数量,应用遗传算法的方式以及所要达到的目标。

总之,标准变异和交叉只是更一般的探测算子的两种方式,现在交叉和变异之间的区分是否必要尚不清楚。无论如何,设计更一般的算子将是一条可行之路。

### 2. 算法对比

模拟自然进化过程可以产生鲁棒的计算机算法——进化算法。在一个统一的框架下对遗传算法、进化规划和进化策略进行比较,可以发现三种算法既有许多相似处,同时也有很大的不同。

进化规划和进化策略都把变异作为主要的搜索算子,而在标准的遗传算法中,变异只处于次要地位;另一方面,交叉在标准遗传算法中起着重要作用,而在进化规划中被完全省去,在进化策略中与自适应结合在一起使用非常重要。另外,标准遗传算法和进化规划都强调随机选择机制的重要性,而从进化策略的角度看,选择是完全确定的,没有合理的根据表明随机选择原则的重要性。进化规划和进化策略确定地把某些个体排除在被选择复制之外,而标准遗传算法一般对每个个体都指定一个非零选择概率。

将来研究的一个明确目标将是探明这一事实的原因,从而为设计新的和更好的进化算法找出一般的原则。

### 3. 应用领域

进化计算是模拟自然界生物演化过程产生的随机优化策略与技术,由于它效率高、易于操作、简单通用,并具有自组织、自适应、自学习等智能特征,因而广泛应用于各种不同的领域。

（1）复杂问题的优化。当所要解决的问题具有非线性、多峰值、不确定性等特征时，使用传统的优化方法常常不能奏效。进化计算由于是一种“黑箱”技术，不要求有明确的因果关系数学表达式，因此它是解决这类问题的有力工具。

（2）复杂系统分析。应用进化计算从事聚类分析、模式识别、图像处理、调度组织等工作，可将表面杂乱无章的复杂事物条理化。

（3）自动控制。进化计算技术具有自适应、自学习、自组织的智能行为，能够适应环境变化，减少波动，保证高的控制精度，保证控制的实时性和快速性。

（4）硬件自动设计。随着 20 世纪 90 年代初易于重构的硬件——现场可编程门阵列（FPGA）的出现，演化硬件（EHW）在国际上受到越来越多的关注，它有可能对复杂电路设计和自适应硬件领域提供一套全新的方法。

（5）自动程序设计。基于遗传程序设计而开展的自动程序设计方法，正在发展成进化软件的研究，即不必精确地告诉计算机具体怎样去做，而由计算机自动完成。

（6）综合应用。进化计算和其他技术相结合，各自发挥特长，综合解决问题。例如，将遗传算法和人工神经网络相结合，解决了机器学习等问题。

## 习　题　10

1. 遗传算法中的染色体与基因是如何表示的？
2. 遗传算法的处理流程是怎样的？
3. 遗传算法中如何确定适应值函数？
4. 选择算子有几种？各自的计算方法是什么？
5. 交叉算子有几种？各自的操作方法是什么？
6. 变异算子有几种？各自的操作方法是什么？
7. 遗传算法的特点有哪些？
8. 根据遗传算法的简例，说明三个遗传算子的作用是什么？
9. 遗传分类学习系统 GCLS 规则生成过程的示意图是什么？
10. GCLS 系统的信任分配算法的步骤是什么？
11. GCLS 系统的遗传算法的主要步骤是什么？
12. GCLS 系统用于脑出血、脑血栓疾病诊断的个体编码方式是什么？
13. 进化计算包括哪些算法？进化算法的特点是什么？
14. 进化策略怎样体现了变异操作的特点？
15. 说明进化规划的计算流程及变异算子的计算公式。
16. 遗传程序设计的基本思想是什么？
17. 如何理解“变异”和“交叉”算子的作用？
18. 对比遗传算法和进化策略与进化规划的不同。

# 第11章 公式发现

## 11.1 公式发现概述

### 11.1.1 曲线拟合与发现学习

在科学发展史上，各种物理学、化学、天文学中的自然规律都是著名科学家对大量的实验数据进行深入的研究，最后得到了自然规律，如牛顿三大定律、万有引力定律、开普勒行星运行定律等。这些自然定律是科学发展和社会进步的奠基石。

自然界存在着无数的规律，除了已被发现的外，还有很多规律需要人们去继续发现。在大量的工程问题中，同样存在着大量的实验数据需要人们去寻找它们的规律性。在找到完全精确的规律性之前，一般用经验性规律（带有一定的误差）来代替，去完成工程计算、设计和施工。经验规律的发现一般是由有经验的工程师来完成的。

**1. 数值计算方法中的曲线拟合**

随着计算机的出现，发展了数据拟合技术。它是数值计算的重要分支。数据拟合是利用科学试验中得出的大量测量数据，去求得自变量和因变量的一个近似公式。

例如，已知 $N$ 个点$(x_i, y_i)$去求得自变量 $x$ 和因变量 $y$ 一个近似表达式 $y=\phi(x)$。

曲线拟合问题的特点在于，被确定的曲线原则上并不特别要求真正通过给定的点，只要求它尽可能从给定点的附近通过。对于含有观测误差的数据来说，不过点的原则显然更为适合。因为它可以部分抵消数据中含有的观测误差。给出它们一般的近似的数学公式有

$$y^* = a_0 + a_1\phi_1(x) + a_2\phi_2(x) + \cdots + a_k\phi_k(x) \tag{11.1}$$

在曲线拟合中，$\phi_k(x)$一般取 $x^k$ 或者是正交多项式。其中 $a_0, a_1, \cdots, a_k$ 各个系数的确定常用的是最小二乘法，即使各点的误差平方和最小：

$$\begin{aligned}\phi(a_0, a_1, \cdots, a_k) &= \sum_1^n (y_i - y_i^*)^2 = \sum_1^n (y_i - (a_0 + a_1\phi_1(x_i) + a_2\phi_2(x_i) + \cdots + a_k\phi_k(x_i)))^2 \\ &= \min\end{aligned} \tag{11.2}$$

对于如何选择 $a_0, a_1, \cdots, a_k$ 使误差平方和最小，可以用数学分析中求极值方法，即函数 $\phi(a_0, a_1, a_2, \cdots, a_k)$对 $a_0, a_1, \cdots, a_k$ 求偏微商，再使偏微商等于零，得到 $a_0, a_1, \cdots, a_k$ 应满足的方程：

$$\begin{cases}\partial\phi/\partial a_0 = -2\sum\limits_{i=1}^N (y_i - a_0 - a_1\phi_1(x_i) - \cdots - a_k\phi_k(x_i)) = 0 \\ \partial\phi/\partial a_1 = -2\sum\limits_{i=1}^N (y_i - a_0 - a_1\phi_1(x_i) - \cdots - a_k\phi_k(x_i)) \cdot \phi_1(x_i) = 0 \\ \partial\phi/\partial a_k = -2\sum\limits_{i=1}^N (y_i - a_0 - a_1\phi_1(x_i) - \cdots - a_k\phi_k(x_i)) \cdot \phi_k(x_i) = 0\end{cases} \tag{11.3}$$

求得这组方程的解$\{a_i\}$,即可得拟合公式(11.1)。

用多项式做逼近公式:

$$y = a_0 + a_1x^1 + a_2x^2 + \cdots + a_kx^k \tag{11.4}$$

根据数学定理,$k$ 越大($x^k$ 的次数越高),逼近的精度越高。但实际计算表明,$k$ 过大,不但求解过程中容易发生病态等麻烦情况,而且得到的多项式尽管在各 $x_i$ 处的值与 $y_i$ 很接近,但其他地方却产生不合理的波动现象。

为克服这方面的困难,取更一般的情况,即用正交多项式 $\Phi_k(x)$代替 $x^k$,它本身是 $k$ 次多项式。典型的如勒让德多项式。用一个例子来说明。

例如,在某一个化学反应里,根据实验所得分解生成物的浓度与时间的关系如表 11.1 所示。

**表 11.1　浓度与时间的关系数据**

| 时间 $t$ | 0 | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 浓度 $y$ | 0 | 1.27 | 2.16 | 2.86 | 3.44 | 3.87 | 4.15 | 4.37 | 4.51 | 4.60 | 4.66 |

由于用简单的多项式作逼近公式,得不到理想的精度,采用勒让德多项式来做逼近公式。在此,用 5 次正交勒让德多项式作为 $y$ 的近似公式:

$$y = \phi_5(x) = \sum_{i=0}^{5} a_i p_{i,10}(x)$$

其中 $x=t/5$,即 $x_0=0, x_1=1, \cdots, x_{10}=10$。

利用曲线拟合方式得到具体的逼近公式为

$$\phi_5(x) = 3.2627 \times 10^{-4} p_{0,10}(x) - 2.15455 \times 10^{-4} p_{1,10}(x) - 0.908104 \times 10^{-4} p_{2,10}(x) - 0.164 \times 10^{-4} p_{3,10}(x) - 0.0195 \times 10^{-4} p_{4,10}(x) - 0.0102 \times 10^{-4} p_{5,10}(x)$$

其中各正交多项式为

$$p_{0,10}(x) = 1$$

$$p_{1,10}(x) = 1 - 2 \cdot \frac{x}{10}$$

$$p_{2,10}(x) = 1 - 6 \cdot \frac{x}{10} + 6 \cdot \frac{x(x-1)}{10(10-1)}$$

$$p_{3,10}(x) = 1 - 12 \cdot \frac{x}{10} + 30 \cdot \frac{x(x-1)}{10(10-1)} - 20 \cdot \frac{x(x-1)(x-2)}{10(10-1)(10-2)}$$

$$p_{4,10}(x) = 1 - 20 \cdot \frac{x}{10} + 90 \cdot \frac{x(x-1)}{10(10-1)} - 140 \cdot \frac{x(x-1)(x-2)}{10(10-1)(10-2)} + 70 \cdot \frac{x(x-1)(x-2)(x-3)}{10(10-1)(10-2)(10-3)}$$

$$p_{5,10}(x) = 1 - 30 \cdot \frac{x}{10} + 210 \cdot \frac{x(x-1)}{10(10-1)} - 560 \cdot \frac{x(x-1)(x-2)}{10(10-1)(10-2)} + 630 \cdot \frac{x(x-1)(x-2)(x-3)}{10(10-1)(10-2)(10-3)} - 252 \cdot \frac{x(x-1)(x-2)(x-3)(x-4)}{10(10-1)(10-2)(10-3)(10-4)}$$

该逼近公式的精度是很高的，但遗憾的是，此公式太复杂，计算起来烦琐，很难理解变量之间的内在关系。

曲线拟合中如何选取基函数(如勒让德多项式)的有效方法是正交筛选法。

可以说，曲线拟合方法基本上解决了在科学与工程中的大量实验数据中找出逼近公式，达到给定的精度。

数据拟合方法虽然能解决一些实际问题，但是它把寻找公式的范围限制在多项式形式之内。对正交多项式一般表示都很复杂，如勒让德多项式，它是由多个多项式组成的。每个多项式的系数都不相同，且多项式次数逐渐增加。由正交多项式表示的逼近公式对使用者来说很不直观，建立不起各个变量之间的直观概念。

**2. 发现学习**

随着人工智能技术的发展，近10年来，机器发现技术得到发展。比较典型的系统有科学定律发现系统BACON、数学概念发现系统AM等。它们都产生了巨大的影响。

对于科学发现的自然规律，用数据拟合的方法在计算机上是绝对得不出来的。只能采用新的途径，这就需要用人工智能技术来完成。BACON系统就是在这种思想指导下产生的。

发现学习是从一组观测结果或数据利用启发式求出这些数据的一个或多个规律。

例如容器中的气体，人们能够观察到的具体数据是温度($T$)、体积($V$)、压强($P$)和克分子个数($N$)。它们之间的规律性是这些属性项之间的关系式：$PV/NT=$常数。公式发现就是找出能够解释给定数据集合的最本质的规律性。

发现学习有两种方式：数据驱动方式的公式发现和模型驱动方式的概念发现。

数据驱动方式的公式发现是根据在搜索数据中所发现的数据规律性，采用不同的启发式发现动作，在一系列发现动作之后形成所发现的公式规律。BACON系统和FDD系统是数据驱动的公式发现系统。

模型驱动方式的概念发现的典型例子是数学概念发现系统AM。它包括了各种各样的搜索法(242个启发式规则)指导在数据领域中的搜索，从集合、表、项等1000多个基本数学概念出发，AM使用具体化、一般化、类比、复合等操作去产生新的数学概念，如得出自然数、质数等重要的数学概念。AM系统还找到了与这些概念有关的定性规律，如唯一因子分解定理等。

## 11.1.2 启发式与数据驱动启发式

**1. 启发式**

启发式是人工智能的重要方法。启发式的基本定义是：能够建议合情的行动和避免不合情的行动的知识。

通过深入的研究，对启发式有了更深入的了解。形成了对启发式的新观点：

(1) 通过使用启发式规则，能开发新的知识领域

通过使用这些既能建议合情的行动，又能排除不合情的行动的启发式集，可以发现一些

全新的概念及其关系。

(2) 当新的知识领域产生和演变时,需要新的启发式

当引入一些新的建议、定理、技术、规范或观察到的现象后,这一领域可能随之改变,用于处理这一领域的启发式也会变化。例如,观察一个用于制定从旧金山到伦敦的旅行计划的启发式集,近些年来,加入了许多新规则,而修改了许多旧规则。

(3) 能用启发式开发新的启发式

启发式本身的生长通过启发式来引导。为了做到这一点,需要很多类型的启发式(如一般的或专用的等)、用于启发式的知识表示以及关于启发式属性的假设等。

(4) 当新的知识领域产生和进化时,需要新的知识表示。新的知识表示也能由启发式产生。

#### 2. 数据驱动启发式

典型的 BACON 系统采用了数据驱动启发式,通过启发式搜索发现科学定律(公式)。

公式发现在于分析数据(或称观测值)得出假说(或称定律)。这些假设(定律)能够解释(或概括)这些数据。

信息用不同层次的描述表示,其中最底层的可认为是数据。而最高层的可说成是假说,中间层次则是这两个概念的混合。一个层次的描述既作为它下面一层描述的假设,又作为它上面一层描述的数据。

BACON 的启发式搜索,总是注意两个数值变量之间增加和减少的单调关系。考察下面一条递减关系的启发式,它可叙述为:如果在某层次的描述中,因变量 $y$ 的值随变量 $x$ 的值减少而增加,则注意 $y$ 和 $x$ 之间的单调减少关系,并计算 $y$ 关于 $x$ 的斜率。一旦某种趋向被发现,系统就计算出有关这两变量组成直线的斜率,即如果发现 $y$ 是 $x$ 的线性函数,其斜率为 $m$,截距为 $i$,BACON 就建立一个斜率变量,定义为 $(y-i)/x$,和一个截距变量为 $y-mx$。

如果截距很接近于零值,BACON 就定义一个比率变量 $y/x$。

如果斜率是常数,那么,系统就建立两个新的变量($m$ 和 $i$),用来定义有关变量的线性组合。

如果该斜率是变化的(它们的关系是非线性的),那么 BACON 就根据关系的方向和所涉及的数的符号去计算有关变量的积或商,系统把这一乘积或商也同样作为一个新变量对待,一旦定义了一个新变量,它都作为变量,再去发现更新的变量关系。

## 11.2 科学定律重新发现系统

### 11.2.1 BACON 系统基本原理

#### 1. BACON 系统的思想

BACON 系统是运用人工智能技术从试验数据中寻找其规律性比较成功的一个系统,

它是 Pat Langley 于 1980 年研制的。它运用数据驱动方法，即这种方法使用的规则空间与假设空间是分开的。系统的规则空间包括若干精炼算子，通过精炼算子修改假设。所谓精炼算子就是修改假设空间的子程序，每个精炼算子以特定的方式修改假设空间。整个学习程序由多个精炼算子组成，程序使用探索知识对提供的训练例进行分析，决定选用哪个精炼算子。这类学习方法的大致步骤如下：

步骤 1　收集某些训练例。

步骤 2　对训练例进行分析，决定应该使用的精炼算子。

步骤 3　使用选出的算子修改当前的假设空间。

重复执行步骤 1 到步骤 3，直到取得满意的假设为止。

BACON 系统的思想是程序反复地考察数据并使用精炼算子创造新项，直到创造的这些项中有一个是常数时为止。于是一个概念就用“项＝常数”的形式表示出来，其中项为变量运算的组合而形成的表达式。

#### 2. BACON 系统主要精炼算子

BACON 系统主要精练算子如下。

（1）发现常数

当某一属性变量取某一值至少两次的时候，触发这个算子，该算子建立这个变量等于常数的假设。

（2）具体化

当已经建立的假设同数据相矛盾时触发这一算子时，它通过增加合取条件的形式把假设具体化。

（3）斜率和截距的产生

当发现两个变量是线性相互依赖时触发这一算子时，它是建立线性关系的斜率和截距作为新变量。

（4）乘积的产生

当发现两个变量以相反方向递增但又不线性依赖时触发该算子时，产生两个变量的乘积作为新变量。

（5）商的产生

当发现两个变量以相同方向递增但又不线性依赖时触发该算子，产生两个变量的商作为新变量。

（6）模 $n$ 变量的产生

当发现两个变量 $v_1$ 和 $v_2$ 在模某一数 $n$ 相等时触发这一算子，产生 $v_2\,(\mathrm{mod}\ n)$ 作为新变量。

### 11.2.2　BACON 系统实例

#### 1. 开普勒第三定律的发现

太阳系行星运行数据包括行星运动周期 $p$（绕太阳一周所需时间），行星与太阳的距离

$d$(绕太阳旋转的椭圆轨道的长半轴),在此用参照数据,以水星数据为单位标准,如表 11.2 所示。

**表 11.2　行星运行数据**

| | $p$ | $d$ | | $p$ | $d$ |
|---|---|---|---|---|---|
| 水星 | 1 | 1 | 地球 | 27 | 9 |
| 金星 | 8 | 4 | | | |

利用 BACON 精炼算子发现行星运行规律过程如表 11.3 所示。

**表 11.3　行星运行规律发现过程**

| | $p$ | $d$ | $d/p$ | $d^2/p$ | $d^3/p^2$ |
|---|---|---|---|---|---|
| 水星 | 1 | 1 | 1 | 1 | 1 |
| 金星 | 8 | 4 | 0.5 | 2 | 1 |
| 地球 | 27 | 9 | 0.33 | 3 | 1 |

发现过程说明如下:

(1) 变量 $p$ 和变量 $d$ 都是递增的,建立两变量相除的新变量 $d/p$(第 3 列)。

(2) 变量 $d$ 与变量 $d/p$ 以相反方向递增,建立两变量相乘的新变量 $d^2/p$(第 4 列)。

(3) 变量 $d/p$ 与变量 $d^2/p$ 以相反方向递增,建立两变量相乘的新变量 $d^3/p^2$(第 5 列)。

(4) 最新变量 $d^3/p^2$ 是常数 1,发现公式为

$$d^3/p^2 = 1$$

### 2. 理想气体定律的发现

理想气体有 4 个变量:体积($V$)、压强($P$)、温度($T$)和克分子个数($N$),具体数据如表 11.4 所示。

**表 11.4　理想气体数据**

| | $V$ | $P$ | $T$ | $N$ |
|---|---|---|---|---|
| $I_1$ | .0083200 | 300 000 | 300 | 1 |
| $I_2$ | .0062400 | 400 000 | 300 | 1 |
| $I_3$ | .0049920 | 500 000 | 300 | 1 |
| $I_4$ | .0085973 | 300 000 | 310 | 1 |
| $I_5$ | .0064480 | 400 000 | 310 | 1 |
| $I_6$ | .0051584 | 500 000 | 310 | 1 |
| $I_7$ | .0088747 | 300 000 | 320 | 1 |
| $I_8$ | .0066560 | 400 000 | 320 | 1 |

续表

| | $V$ | $P$ | $T$ | $N$ |
|---|---|---|---|---|
| $I_9$ | .0053248 | 500 000 | 320 | 1 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| $I_{25}$ | .0266240 | 300 000 | 320 | 3 |
| $I_{26}$ | .0199680 | 400 000 | 320 | 3 |
| $I_{27}$ | .0159740 | 500 000 | 320 | 3 |

为了发现它们之间的规律，先取变量 $T$ 和 $N$ 的相同的数据（如前三列中 $T=300$，$N=1$），对变量 $V$ 和 $P$ 进行分析发现，由于 $V$、$P$ 两变量以相反方向递增，利用 BACON 精练算子，建立两变量相乘的新变量 $PV$，且 $PV$ 等于常数 2496。

对于另一组相同的数据（$T=310$，$N=1$），利用相同方法得到 $PV$ 新常数 2579.1999。这样得到新的理想气体数据，如表 11.5 所示。

**表 11.5　合并 *PV* 变量后的理想气体数据**

| | $PV$ | $T$ | $N$ |
|---|---|---|---|
| $I_1'$ | 2 496 | 300 | 1 |
| $I_2'$ | 2 579.1999 | 310 | 1 |
| $I_3'$ | 2 622.3999 | 320 | 1 |
| $I_4'$ | 4 991.9999 | 300 | 2 |
| $I_5'$ | 5 158.3999 | 310 | 2 |
| $I_6'$ | 5 324.7999 | 320 | 2 |
| $I_7'$ | 7 488 | 300 | 3 |
| $I_8'$ | 7 737.5999 | 310 | 3 |
| $I_9'$ | 7 987.2 | 320 | 3 |

从表 11.4 到表 11.5，合并了变量 $P$ 和 $V$ 成新变量 $PV$，它和变量 $T$ 和 $N$ 仍是三个变量。为了有效地发现它们之间的规律，仍先固定变量 $N$，研究变量 $PV$ 与 $T$ 之间的关系。表 11.5 中每三行数据均为 $N=1$、2、3 是常数的数据。

分析在 $N=$常数的三行数据中，变量 $PV$ 与 $T$ 是以相同方向递增，利用 BACON 精炼算子，建立两变量相除的新变量 $PV/T$，且新变量等于常数（取不同的 $N$ 时，$PV/T$ 常数不同）。这样得到的理想气体数据如表 11.6 所示。

**表 11.6　最新的理想气体数据**

| | $PV/T$ | $N$ | | $PV/T$ | $N$ |
|---|---|---|---|---|---|
| $I_1''$ | 8.32 | 1 | $I_3''$ | 24.95 | 3 |
| $I_2''$ | 16.64 | 2 | | | |

对表 11.6 中数据，它是两变量 $PV/T$ 与 $N$ 的数据。分析两变量 $PV/T$ 与 $N$ 的变化关系。两变量以相同方向递增，利用 BACON 精炼算子，建立两变量相除的新变量 $PV/T/N = PV/TN$，得到常数 8.32，按 BACON 精炼算子，发现公式为

$$PV/NT = 8.32$$

BACON 系统在发现某些科学定律上取得很大成功，但是 BACON 系统也存在很多弱点。第一个弱点是 BACON 系统对训练例所取得的具体值特别敏感，产生这种情况的原因是每一个精炼算子都有十分具体的触发条件，训练例的值一变，或者提供训练例的次序一变，都会影响规则的触发。例如，对某一类训练例 BACON 不能发现欧姆定律，如果变量的次序安排得不够好，BACON 发现单摆定律要多花 40%的时间。

第二，BACON 不能处理干扰性的训练例。例如，发现常数的精炼算子的触发仅仅是根据某一项在两个训练例的值相等。这种触发条件显然对干扰是高度敏感的。

### 11.2.3 BACON 系统的进展

BACON 系统共有 5 个版本，不同的版本其规则空间也不同。

(1) BACON.1 提出了 6 条精练算子，发现了开普勒定律。

(2) BACON.2 是 BACON.1 的扩展形式，它包括两条附加的运算程序，能够发现递归序列并通过计算重复差的方法产生多项式，BACON.2 的能力有很大提高，可以解决一大类序列外推的任务。

(3) BACON.3 是 BACON.1 的另一扩展形式，它使用发现常数运算程序提出的假设重新构造训练例。它用不同的描述层次来表示数据，其中最低层是直接观察的，最高层对应于数据的假说，中间层相对于下层，它是假说，相对于上层它是数据，它不把假说和数据截然分开。BACON.3 由大约 86 个产生式规则组成，共分 7 组，各组产生式规则负责不同的任务，有的负责直接搜索观测数据，有的负责数据的规律性，有的计算项的值，有的把新项分解为它的组成部分。

BACON.3 发现的规律有：

理想气体定律：$pv/(nt)=k_1$

Coulomb 定律：$fd^2/(q_1q_2)=k_4$

Galileo 定律：$dp^2/(lt)^2=k_5$

Ohm 定律：$td^2/(l_c-k_6c)=k_7$

(4) BACON.4 把观察变量的组合式认为是推理项，它使用了启发式搜索方法：程序总是注意两个数值变量之间增加和减少的单调关系，如果斜率为常数，则系统建立两个新的推理项(斜率项和截距项)作为有关变量的线性组合。如果斜率是变化的(不是线性关系)，则 BACON.4 计算有关项的乘积或比值，并把这个变量当做一个新的推理项，一旦新的项确定了，就不需区别推理项和观察变量。BACON.4 递归应用同样试探规则，使系统具有相当大的发现经验规律的能力。该系统还提出了固有性质解决符号变量的处理。

BACON.4 又发现了若干自然规律：

Snell 折射定律：$\sin(i)/\sin(r)=n_1/n_2$

动量守恒动量：$m_1v_1=m_2v_2$

万有引力定律：$F=Gm_1m_2/d_2$

Black 比热定律：$c_1m_1t_1+c_2m_2t_2=(c_1m_1+c_2m_2)t_f$

(5) BACON.5 用简单的类比推理发现守恒定律，对两个物体具有完全相同的有关项，BACON.5 推测最后的定律是对称的。它把各项排序，使得属于同一物体的项首先改变，一旦该物体的这些变量中发现一个不变推理项，程序就假定必有一个类似项可用于另一物体。因此，BACON.5 只需相同地改变另一个项集合中的推理项。当做到了这一点之后，两个高层项取不同的值，可用其他试探规则查找它们之间的关系。这样一来，在物理中普遍存在的对称定律可以很容易地发现。

BACON.5 发现了能量守恒定律。

# 11.3 经验公式发现系统

## 11.3.1 FDD 系统基本原理

经验公式发现系统 FDD(Formula Discovery from Data)是陈文伟团队应用人工智能技术的机器发现技术和数值计算中的曲线拟合技术以及可视化技术结合起来自行研制的系统。它是从大量试验数据中发现经验公式。逐步完成任意函数的任意组合(线性组合、初等运算组合、复合函数运算组合等)，对自然规律和经验规律的发现。

FDD 系统有三个版本：FDD.1，FDD.2，FDD.3。

FDD.1 系统能够发现变量取初等函数或复合函数的组合公式。FDD.2 系统能够发现变量取导数的公式。FDD.3 系统能发现多变量取初等函数或复合函数的组合公式。

### 1. 问题描述

给定一组可观察变量 $X(x_1, x_2, \cdots, x_n)$ 以及这组变量的试验数据 $D_i(d_{i1}, d_{i2}, \cdots, d_{in})$，$i=1,2,3,\cdots,m$ 公式发现系统找出该组变量满足的数学关系式：$f(x_1, x_2, \cdots, x_n)=c$，其中 $c$ 为常数，亦即对于任意一组试验数据$(d_{i1}, d_{i2}, \cdots, d_{in})$均满足关系式 $f(d_{i1}, d_{i2}, \cdots, d_{in})=c$。

所找出的关系式 $f(x)$ 是任何形式的数学公式，包括分段函数。

对于关系式 $f(x_1, x_2, \cdots, x_n)=c$ 的复杂程度可分为：

(1) 变量的初等运算：$f(x,y)=x\theta y$，其中 $\theta$：+、-、*、/。

(2) 变量的初等函数运算：$f(x)=c$，其中 $f(x)$为初等函数。

(3) 初等函数的任意组合：$f(x,y)=a_1f(x)\theta a_2f(y)$。

(4) 复合函数的运算 $g(f(x))=c$，其中 $g(x)$、$f(x)$均为初等函数。

(5) 复合函数的任意组合：$h(a_1g_1(f(x))\theta a_2g_2(f(y)))$，其中 $h(x)$、$g(x)$、$f(x)$均为初等函数。

(6) 多个初等函数的组合：$f(x,y)=a_1f_1(x)\theta a_2f_2(x)\cdots\theta a_kf_k(y)$，其中 $f(x)$、$f(y)$均为初等函数。

(7) 分段函数：对于不连续的点，分别用不同的函数加以描述。

以上是对两个变量的讨论。在现实世界中存在着多变量的更为复杂的关系，在公式发

现过程中采用先寻找两变量的关系，再逐步扩充为多变量的关系的方法。

### 2. FDD.1 的设计思想

FDD.1 系统的基本思想是利用人工智能启发式搜索函数原型、寻找具有最佳线性逼近关系的函数原型，并结合曲线拟合技术及可视化技术来寻找数据间的规律性。

启发式方法是求解人工智能问题的一个重要方法。一般启发式是建立启发式函数，用以引导搜索方向，以便用尽量少的搜索次数，从开始状态达到最终状态。

FDD.1 系统在执行搜索的过程中，对原型函数的搜索以及对它们的组合函数的搜索，也是一种组合爆炸现象。为解决这一问题，在设计系统时采用了启发式方法来实现。

对某一变量取初等函数和另一变量的初等函数或基本数据进行线性组合，即从原型库中选取逼近效果最好的少数几个初等函数作为基函数，并进一步形成组合函数，直至找到最后的目标函数。FDD.1 系统的启发式函数形式为

$$f(x_2) = a + bf_1(x_1) \tag{11.5}$$

线性逼近误差公式为

$$\mathrm{d}t = (a + bf(x_1) - f(x_2))/f(x_2) \tag{11.6}$$

通常总是选取 $\mathrm{d}t$ 最小的 $f(x_i)$ 作为继续搜索的当前结点。这一启发式函数在以后的多次应用中被证明是有效的。

### 3. FDD.1 系统中的知识

在 FDD.1 系统中，知识采用的是产生式规则的表示形式(if…then)。

主要的基本规则有：

**规则 1** 发现常数

当某一变量 $x$ 取一个常数，则建立该变量等于常数的公式，即 $x=c$。

**规则 2** 两变量的初等运算组合

当两变量进行初等运算若等于常数，则建立该变量的初等运算关系式：

$$a_1x_1\theta a_2x_2 = c$$

其中 $\theta$：$+$、$-$、$\times$、$/$。

**规则 3** 变量取初等函数

当某变量取初等函数等于常数，则建立该变量的初等函数关系式：

$$f(x)=c$$

其中 $f(x)$为初等函数。

**规则 4** 两变量取初等函数的线性组合

两变量分别取初等函数后的线性组合等于常数，则建立两变量取初等函数的线性组合关系式：

$$a_1f_1(x_1) + a_2f_2(x_2) = c$$

其中 $f_1(x_1)$、$f_2(x_2)$为初等函数。

**规则 5** 某变量取某一初等函数与另一变量的线性组合

对某一变量 $x_i$ 取初等函数后与另一变量 $x_j$ 进行线性组合，若为常数，则建立关系式：

$$c_1 f(x_i) + c_2 x_j = c$$

**规则 6** 对某一变量 $x_j$ 取初等函数，另一变量 $x_i$ 取两个初等函数进行线性组合，若为常数，则建立关系式：

$$c_1 f_1(x_i) + c_2 f(x_i) + c_3 g(x_j) = c$$

**规则 7** 建立新变量(启发式 1)

若两变量的某初等运算结果接近常数，则建立新变量为该两变量的某种初等运算。

**规则 8** 建立某变量的某种初等函数为新变量(启发式 2)

若某变量的某种初等函数与另一变量或它的初等函数进行线性组合接近常数，则建立该变量的初等函数为新变量。

以上规则的嵌套或递归使用，将形成变量的任意函数间的任意组合。在应用规则时，利用可视化技术将减少各种函数和各种运算的选取，大大节省了搜索时间。

## 11.3.2 FDD.1 系统

### 1. 总体结构图

FDD.1 总体结构图如图 11.1 所示，该系统由试验数据输入、数据生成器、公式发现控制、可视化过程、数据项、原型选择、公式生成、误差分析、循环控制、公式输出与可视化显示十个模块以及原型算法库、试验数据库、知识库、公式库四个库组成。

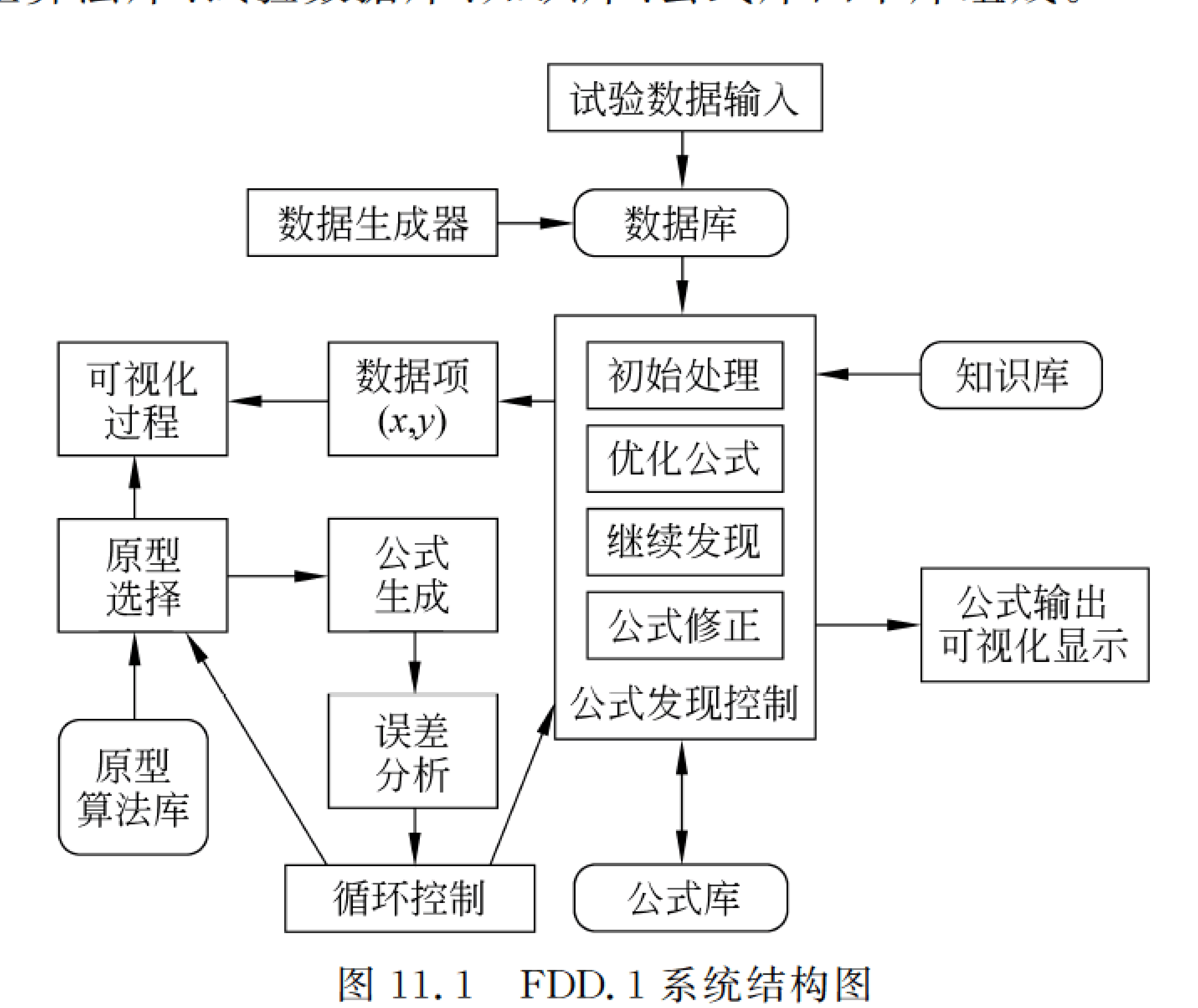



图 11.1 FDD.1 系统结构图

### 2. 各模块说明

(1) 试验数据输入

试验数据输入(Input Data)用于提示用户输入试验数据。

(2) 数据生成器

数据生成器(Generator)用于测试系统效果。给定一个已知公式后，它能生成一批数

据，FDD.1 系统的核心程序将利用这些数据来找出已给定的公式，从而达到测试系统的公式发现能力的效果。此模块是一个可独立执行模块。

（3）数据库

数据库（DataBase）存放待处理的变量数据，一般是科学和工程实验数据。公式的正确与否与数据的规律性和充分性密切相关。系统本身可提供直接输入数据的功能，用户可在系统的提示下将数据输入。也可用数据生成器为系统提供数据，系统将其按一定的格式存储起来，存放在数据库中。数据库中有一个缓冲区，供系统运行时存放中间变量数据以及实现数据的移动和变化。

（4）可视化过程

此模块又分成三个子模块：

- 描绘试验数据的变化趋势。
- 描绘出原型算法库中各函数原型的变化规律。此子模块具有很大的灵活性，用户可根据需要随意调用所选择原型以描绘其变化趋势。
- 描绘所发现的公式的变化规律与原始数据之间误差分布状况。

（5）公式发现控制模块

此模块是 FDD.1 的核心部分，它主要是利用知识库中的知识，优选函数原型、控制继续发现、公式修正等。它包含：初始处理、优选公式、继续发现、公式修正四个子模块。下面对这四个子模块的功能加以说明：

① 初始处理。此模块的主要功能有两个方面，其一是根据具体情况对用户所提供的数据进行初步处理；其二是在多变量中选择两个变量以及向多变量的过渡处理。

② 优选公式。其主要功能是对公式库中提供的公式根据其误差逼近情况来优选函数原型，对函数原型一般选择 2～3 个。

③ 继续发现。此模块将根据误差分析情况完成如下功能：

- 建立新变量。
- 颠倒变量关系。
- 对所选择的函数原型进行组合。

④ 公式修正。这是在输出公式之前所必经的一个过程，此过程将根据用户提供的误差要求决定是否对系统所发现的公式进行修正（说明：此处是对已发现的公式的误差进行修正）。若不必修正则将公式送入“公式输出”与“可视化”模块；否则对公式进行修正。目前系统提供了三种公式修正方法，如下所述：

- 调和级数回归。由数学分析可知，对任意周期函数 $y=f(x)$，可以用三角函数的傅里叶级数来逼近，即：

$$y=\phi(x)=a_0+\sum_{j=1}^{m}(a_j\cos(jx)+b_j\sin(jx)) \tag{11.7}$$

将 $n$ 组试验数据 $(x_i, y_i)$ 代入上式，各点误差值以调和函数方程式的形式表示为

$$\begin{gathered}y_i=a0+\sum_{j=1}^{m}(a_j\cos(jx_i)+b_j\sin(jx_i))\\ i=1,2,\cdots,m;j=1,2,\cdots,m\end{gathered} \tag{11.8}$$

可以按最小二乘原理求出调和级数中各未知系数。

• 用直线来描述误差：

此算法和公式生成模块的直线拟合法类似。

• 神经元网络方法逼近误差函数

利用神经元网络中函数式网络对误差函数进行计算，求出网络权值，使函数型网络逼近该误差函数。函数型网络选取的函数为

$$\sin(2k\pi x)、\cos(2k\pi x) \quad k = 1,2,\cdots,n$$

(6) 数据项

程序中的两个指针变量用以存放在多个变量中所选择出的两个变量的实验数据。

(7) 选择原型

此过程通过调用原型算法库、可视化过程及误差分析模块提供的误差来进行函数原型的选择。有两种选择方式：

• 由用户指定选择。

• 通过循环控制进行顺序选择。

(8) 公式生成

此模块主要应用数值分析中的曲线拟合技术求出拟合公式的系数，同时生成公式。

(9) 误差分析模块

此模块的主要功能是对公式生成模块提供的公式，计算相对误差并对各公式误差进行比较。

(10) 循环控制模块

此模块设有一个控制开关，对“原型选择”和“公式发现控制”两个过程进行循环运行。

(11) 公式输出和可视化显示

此过程是系统所要执行的最后一步，当公式发现控制模块决定最终输出公式后执行此模块，输出公式并进行可视化显示。这样用户就可以很直观地阅读公式、了解所发现的公式逼近实验数据的情况。

(12) 原型算法库

原型是构成数学公式的基本单元，原型算法库所包括的原型决定了系统的发现能力。本系统的函数原型由基本原型和组合原型构成。

基本原型由初等函数组成，如：

$x$、$x^2$、$x^3$、$x^{-1}$、$x^{-2}$、$\mathrm{sqrt}(x)$、$x^{1/3}$、$\log(x)$、$\exp(x)$、$\sin(x)$、$\cos(x)$等。

组合原型由初等函数的初等运算组合而成，如：

$x\sin(x)$、$x\cos(x)$、$x\exp(x)$、$x\log_{10}(x)$、$x^{-1}\log_{10}(x)$、$x^{-1}\exp(x)$、$1/\log_{10}(x)$、$1/\mathrm{sqrt}(x)$、$\sin(x)+\cos(x)$等。

在原型算法库中，每个原型都给出了一个算法，只不过每个算法的程序结构都非常相似。

用户还可以根据需要随意增加、删除原型，在程序运行过程中给出了一个控制参数，用户可通过它来调用所需算法。

(13) 知识库

知识库中知识用于构造和发现关系式。

(14) 公式库

公式库用来存放在系统搜索过程中初步选择的原型函数组成的公式,以备公式发现控制模块使用。

公式库中的公式包含两个变量取某原型函数的线性组合以及该公式的逼近误差。在搜索过程中,每当发现一个比较可行的公式或函数原型,便将其送入公式库等待下一步的选择,每一轮选择之后便把落选的公式剔除出公式库,直至发现满意的公式为止。

### 3. FDD.1 系统实例

(1) 行星运动开普勒第三定律的重新发现

① 原始数据

原始数据如表 11.7 所示。

**表 11.7 行星运行的近似数据**

| 距离 $d$ | 1 | 4 | 9 | 16 | 25 | 36 | 49 | 64 | 81 | 100 |
|---|---|---|---|---|---|---|---|---|---|---|
| 周期 $p$ | 1 | 8 | 27 | 64 | 125 | 216 | 343 | 512 | 729 | 1000 |

② 开普勒第三定律搜索树

对于行星绕太阳运动的开普勒第三定律,BACON 系统利用变量的乘除运算,使得到的新变量趋向常数的思想,对该定律重新发现。利用变量取初等函数的线性组合趋向直线方程的思想,对该定律也重新发现,公式发现的搜索树如图 11.2 所示。从搜索过程可见,FDD.1 系统的公式的发现过程与 BACON 系统的公式发现过程是完全不同的。

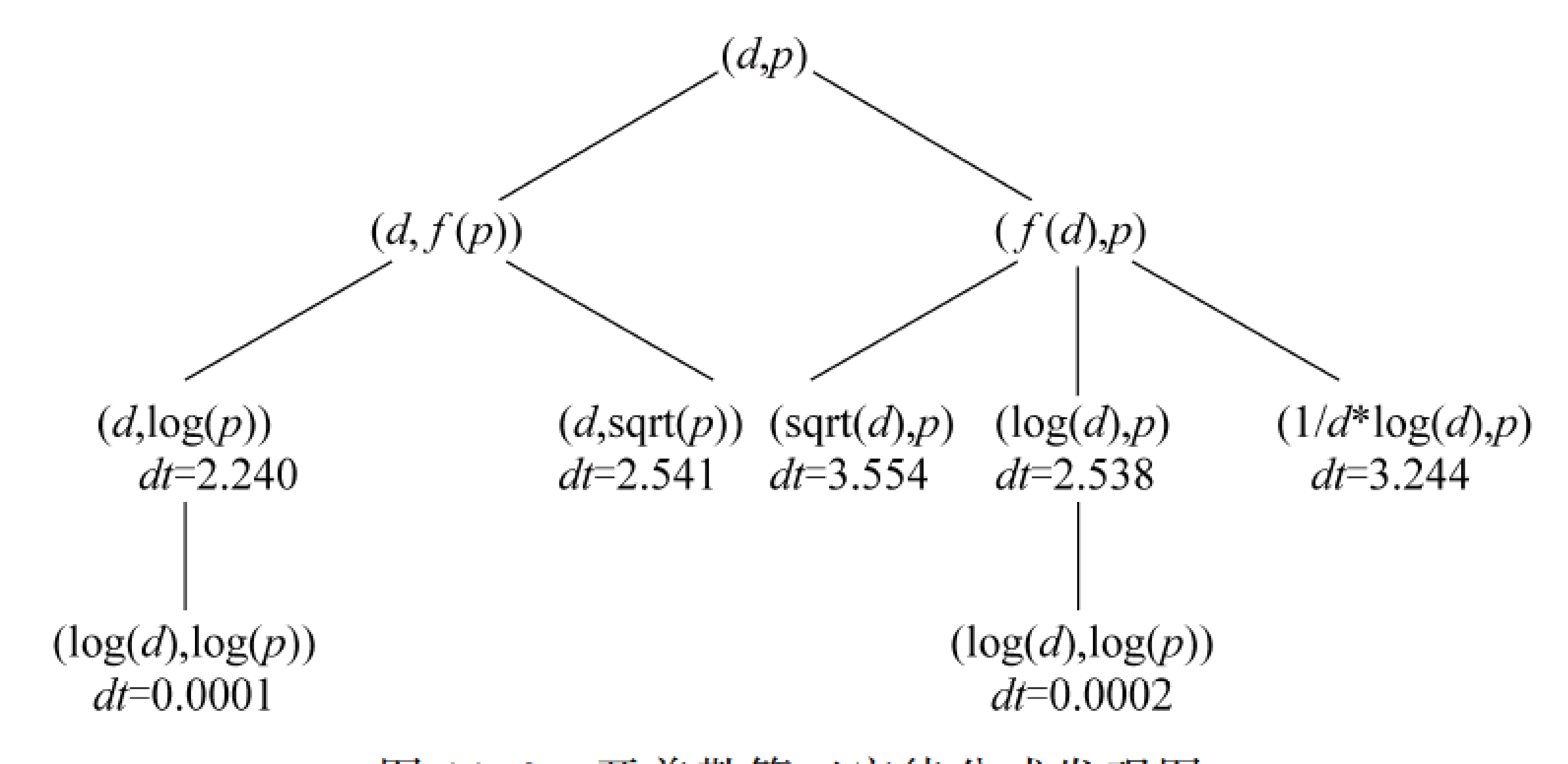



图 11.2 开普勒第三定律公式发现图

公式发现搜索树中有两个分支,左分支路径为:先固定 $d$,对变量 $p$ 求各原型函数 $f(p)$,用 $d$ 和 $f(p)$拟合线性方程 $f(p)=a+bd$,其中 $a$、$b$ 是常数,求逼近 $f(p)$的相对误差,选误差最小的函数为 $\log(p)$,误差为 2.240,建立新变量 $p'=\log(p)$,并固定它,再对 $d$ 变量求各原型函数 $g(d)$,对 $\log(p)$和 $g(d)$拟合线性方程,并求逼近 $g(d)$的相对误差,选取误差最小者为 $\log(d)$,误差为 0.00001,调用公式生成模块求得公式及系数,公式为

$$\log_{10}(d) = 0.0 + 0.666666667 * \log_{10}(p) \tag{11.9}$$

即为

$$d^3 = p^2$$

从右分支树也可发现开普勒第三定律，这里不再详述。

(2) 实例数据的公式发现

例如，炼钢厂出钢时所用盛钢水的钢包，在使用过程中由于钢液及炉渣对包衬耐火材料的侵蚀，使其容积不断增大，钢包的容积与相应的使用次数(即包龄)的数据如表 11.8 所示。

**表 11.8　钢包容积数据**

| 使用次数 $x$ | 容积 $y$ | 使用次数 $x$ | 容积 $y$ |
|---|---|---|---|
| 2 | 106.42 | 11 | 110.59 |
| 3 | 108.20 | 14 | 110.60 |
| 4 | 109.58 | 15 | 110.90 |
| 5 | 109.50 | 16 | 110.76 |
| 7 | 110.00 | 18 | 111.00 |
| 8 | 109.93 | 19 | 111.20 |
| 10 | 110.49 | | |

对这组试验数据的搜索过程与行星运动例子相同，这里不再详细叙述其具体发现过程，只给出它的公式发现搜索树和最终公式形式，并与《计算方法引论》书中方法及结果做比较，公式发现搜索树如图 11.3 所示。

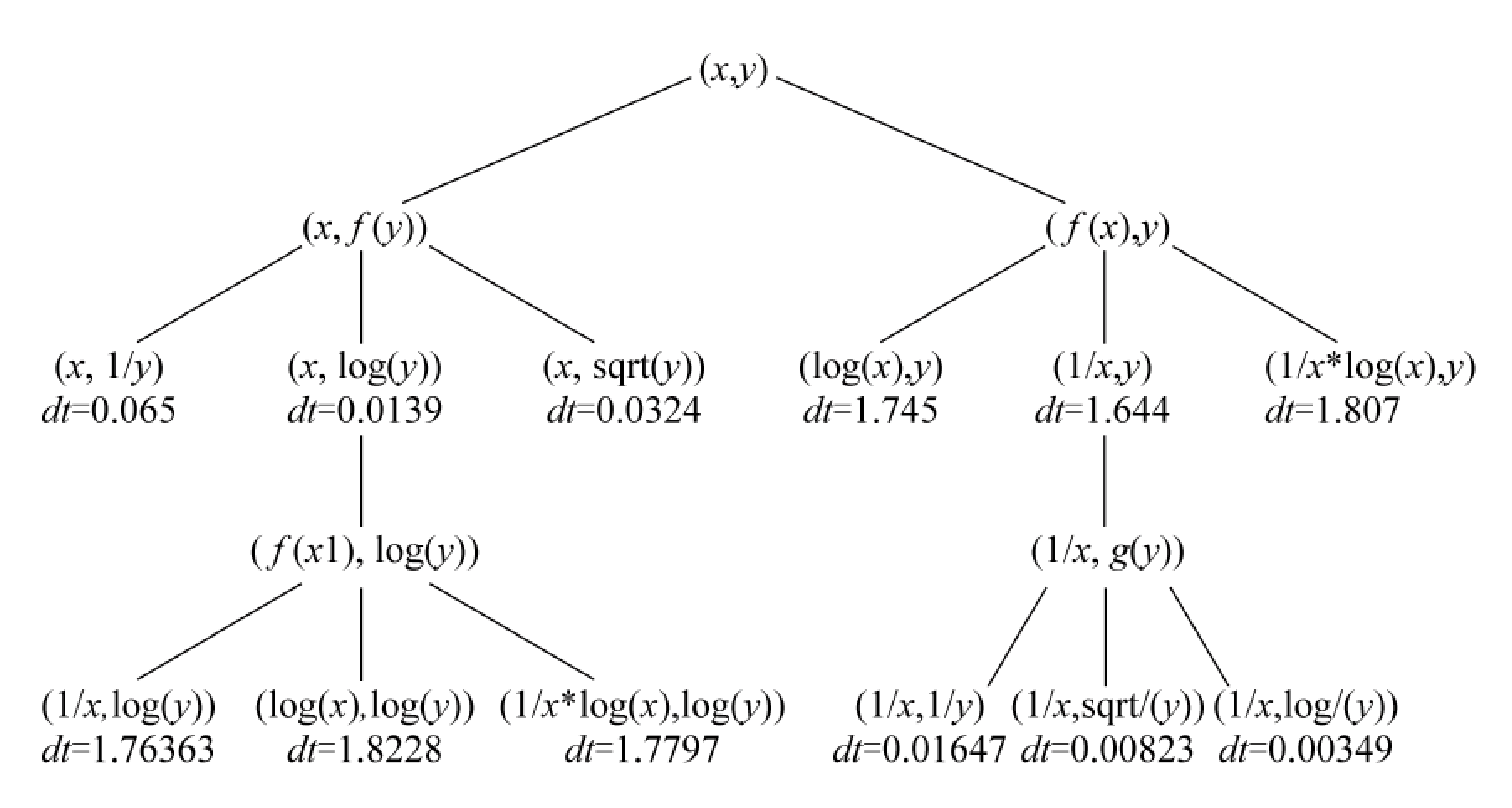



图 11.3　钢包容积变化公式发现图

从右分支开始搜索，得到了组成公式的两组基函数为：(1/$x$, log($y$))；(1/$x$, sqrt($x$))调用公式发现模块求得公式及系数，最终得到经验关系式为

$$\text{sqrt}(y) = 10.5591908 - 0.4711268/x \tag{11.10}$$

$$dt = 0.008233$$

$$\log(y) = 2.0472975 - 0.0392124/x \tag{11.11}$$

$$dt = 0.00349$$

经效果分析均满足误差要求。

这样就用 FDD.1 系统发现了上述两个公式。

《计算方法引论》一书所讲述的公式为

$$y = x/(0.008966 + 0.00083012x) \tag{11.12}$$

这个公式是该书作者根据自己的专业知识和经验，并根据其离散点在图上分布形状选择 $1/x$ 代替 $x$，$1/y$ 代替 $y$，再进行线性拟合而得到的公式。此公式与图 11.3 中得到的 $(1/x, 1/y)$ 作为新变量求得的线性组合公式一样，其误差是 $dt=0.01647$，比前两个公式的误差要大。这说明 FDD 方法能代替人的经验。

从许多试验数据的分布状况中，人们往往看不出它的具体规律，因此利用人的经验的做法不具有普遍性，而且具有一定的盲目性。而使用 FDD.1 发现经验公式并不要求用户的经验和专业知识，FDD.1 系统很快便能发现效果良好的经验公式，这是 FDD 系统的一个显著优点。

## 11.3.3 FDD.2 系统

### 1. FDD.2 问题描述

给定两组可观察变量 $X(x_1, x_2)$ 以及这组变量的实验数据 $D_i(d_{i1}, d_{i2})$，$i=1,2,3,\cdots,n$，公式发现系统找出该组变量满足的数学关系式：$f(x_1, x_2)-c=\min$，其中 $c$ 为常数，亦即对于任意一组实验数据 $(d_{i1}, d_{i2})$，均满足关系式 $f(d_{i1}, d_{i2})-c=\min$，所找出的关系式 $f(x)$ 是任何形式的数学公式。

对于关系式 $f(x_1, x_2)-c=\min$ 中的函数 $f$ 的复杂程度可分为：

- 变量的初等运算 $f(x,y)=x\theta y$，其中 $\theta$：+、-、*、/；
- 变量的初等函数运算 $f(x)=c$，其中 $f(x)$ 为初等函数；
- 初等函数的任意组合 $f(x,y)=a_1 f(x)\theta a_2 f(y)$；
- 复合函数的运算 $g(f(x))=c$，其中 $g(x)$、$f(x)$ 均为初等函数等；
- 导数处理函数。

设给出的测量数据为：

| $I$ | 1 | 2 | … | $N$ |
|---|---|---|---|---|
| $X$ | $x_1$ | $x_2$ | … | $x_n$ |
| $Y$ | $y_1$ | $y_2$ | … | $y_n$ |

则，一阶差分：$\Delta x_k = x_{k+1} - x_k$；$\Delta y_k = y_{k+1} - y_k$；$(k=1,2,\cdots,n-1)$

二阶差分：$\Delta^2 y_k = \Delta y_{k+1} - \Delta y_k$；$\Delta^2 x_k = \Delta x_{k+1} - \Delta x_k$ $(k=1,2,\cdots,n-2)$

……

$m$ 阶差分 $\Delta^m y_k = \Delta^{m-1} y_{k+1} - \Delta^{m-1} y_k$，

在这里差分指向前差分。

一阶差商　$\delta y_k = (y_{k+1} - y_k)/(x_{k+1} - x_k)$　$(k=1,2,\cdots,n-1)$

二阶差商　$\delta^2 y_k = (\delta y_{k+1} - \delta y_k)/(x_{k+2} - x_k)$　$(k=1,2,\cdots,n-2)$

……

$m$ 阶差商 $\delta^m y_k = (\delta^{m-1} y_{k+1} - \delta^{m-1} y_k)/(x_{k+m} - x_k)$

可以用导数表达差商,若 $f(x)$在$[a,b]$上 $n$ 次可微,$x_1,\cdots,x_n$ 是$[a,b]$内的$(n)$个不同的点,则有 $\xi(a<\xi<b)$使 $\delta^{n-1}y = f^{(n-1)}(\xi)/(n-1)!$。

### 2. FDD.2 规则描述

在 FDD.2 系统中,知识同样采用的是产生式规则的表示形式(if…then)。包括 FDD.1 系统的规则外,还包括如下规则:

**规则 1** 差分发现常数

当某一变量 $y$ 的差分取一个常数 $c$,则建立该变量等于常数的公式,即 $y=a+cx$。

**规则 2** 差商发现常数

当两个变量 $y$ 的差商取一个常数 $c$,则建立该变量等于常数的公式,即 $y'=c$。

**规则 3** 特殊函数形式导数函数

(1) 阶差(向前差分)法判定类型

若 $\Delta^2 y_i$=定值,则方程为 $y=a+bx+cx^2$;

若 $\Delta^3 y_i$=定值,则方程为 $y=a+bx+cx^2+dx^3$;

若 $\Delta(y_i)'$=定值,则方程为 $y^{-1}=a+bx$;

若 $\Delta^2(y_i^2)$=定值,则方程为 $y^2=a+bx+cx^2$;

若 $\Delta^2(x_i/y_i)$=定值,则方程为 $y=x/(a+bx+cx^2)$;

若 $\Delta y_i$ 成等比数列,则方程为 $y=ab^x+c$;

若 $\Delta\log(y_i)$成等比数列,则方程为 $\log(y)=a+bx+cx^2$;

若 $\Delta^2 y_i$ 成等比数列,则方程为 $y=ab^x+cx+d$。

(2) 差商判定类型

若 $\Delta\log(y_i)/\Delta\log(x_i)$=定值,则方程为 $logy=ax^b$;

若 $\Delta\log(y_i)/\Delta x_i$=定值,则方程为 $y=ab^x$;

若 $\Delta(x_i y_i)/\Delta x_i$=定值,则方程为 $y=a+b/x$;

若 $\Delta(x_i/y_i)/\Delta x_i$=定值,则方程为 $y=x/(ax+b)$;

若 $\Delta y_i/\Delta(x_i^2)$=定值,则方程为 $y=a+bx^2$。

**规则 4** 变量的导数运算组合

当某变量差分(或差商)后与另一变量进行初等运算若等于常数,则建立该变量差分(或差商)的初等运算关系式:

$$\Delta f(x_1)\theta f(x_2) = c$$

其中 $\theta$:+、-、*、/,其中 $\Delta f$ 为差分或差商计算。

**规则 5** 两变量取导数运算的线性组合

两变量分别取差商运算后的线性组合等于常数 $c$,则建立两变量取导数运算的线性组合关系式:

$$a_1\delta f_1(x_1) + a_2\delta f_2(x_2) = c$$

其中 $\delta f_1(x_1)$、$\delta f_2(x_2)$为差商运算。

以上规则和 FDD.1 中的规则的嵌套或递归使用，将形成变量的任意函数和导数运算组合。

### 3. FDD.2 公式发现实例

(1) 导数函数公式的发现

$x$，$y$ 为样本数据，$Y$ 为发现的公式计算值，如表 11.9 所示。

**表 11.9　导数函数公式的发现**

| $x$ | 1.01 | 2.07 | 2.98 | 7.89 | 7.02 | 6.03 | 6.98 | 8.01 | 9.04 | 9.99 | 11.02 | 12.01 | 12.97 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $y$ | 4.61 | 10.51 | 14.65 | 14.61 | 11.08 | 10.2 | 12.6 | 18.27 | 27.3 | 24.46 | 22.08 | 19.72 | 20.93 |
| $Y$ | 4.667 | 10.662 | 14.248 | 14.524 | 11.741 | 10.383 | 12.679 | 18.263 | 27.174 | 24.257 | 22.045 | 19.965 | 21.115 |

发现导数函数公式：$y'=1.52-4.34\sin(x)$，误差：0.048。

(2) 复合函数公式的发现

数据如表 11.10 所示。

**表 11.10　复合函数公式的发现**

| $x$ | 0.10 | 0.12 | 0.23 | 0.25 | 0.30 | 0.26 | 0.55 | 0.76 | 0.81 | 0.89 | 0.91 | 1.01 | 1.44 | 1.50 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $y$ | 7.146 | 7.288 | 6.156 | 6.329 | 6.782 | 6.417 | 9.532 | 12.588 | 17.443 | 14.936 | 17.337 | 17.53 | 37.81 | 47.02 |
| $Y_1$ | 4.899 | 5.044 | 5.924 | 6.10 | 6.561 | 6.190 | 9.371 | 12.51 | 13.385 | 14.921 | 15.334 | 17.59 | 36.50 | 43.98 |
| $Y_2$ | 6.65 | 6.66 | 6.67 | 6.80 | 6.92 | 6.82 | 8.38 | 11.236 | 12.204 | 14.021 | 14.530 | 17.43 | 37.91 | 41.98 |
| $Y_3$ | 7.185 | 7.310 | 6.07 | 6.228 | 6.636 | 6.306 | 9.268 | 12.525 | 17.491 | 17.223 | 17.696 | 18.33 | 37.0 | 40.99 |

发现公式：$Y_1=7.94x-11.64\log(|\cos(x)|)+4.25$ 公式的误差为 0.095，如图 11.4 所示。

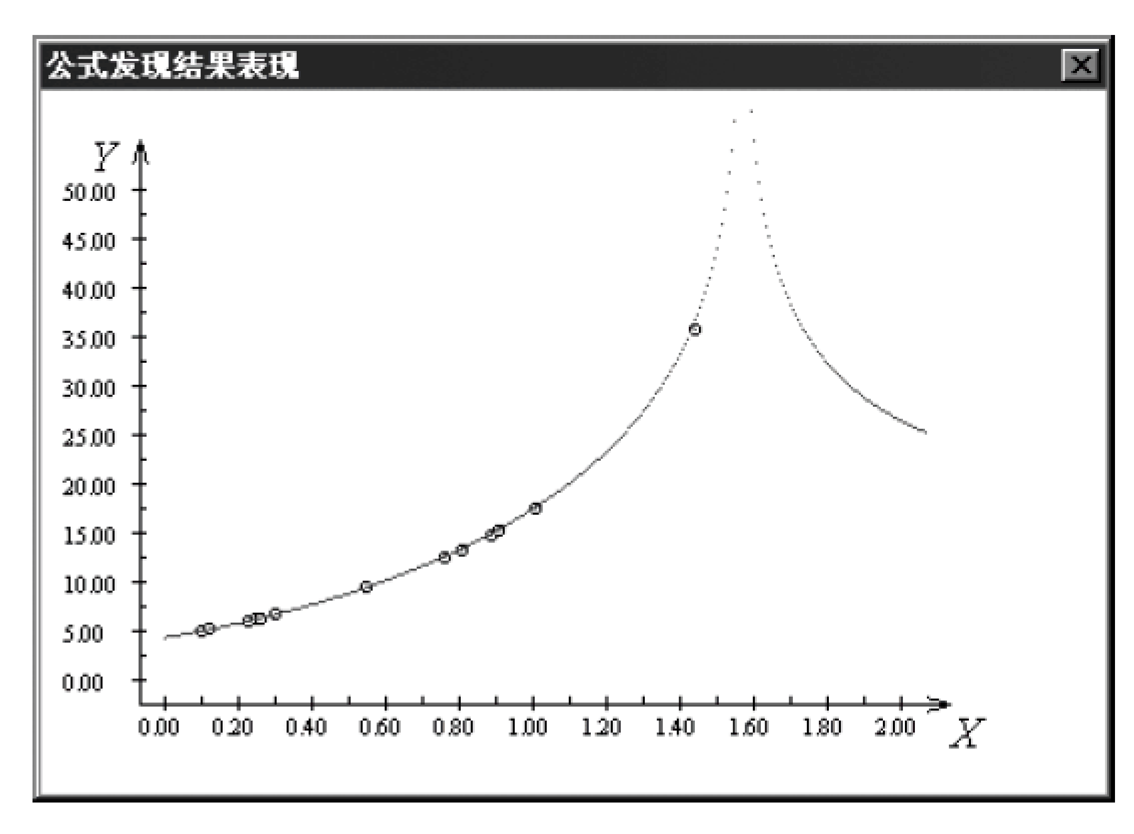



图 11.4　复合函数公式发现

另外还发现两个公式：

$$Y_2 = 6.639005246 + 10.47187751x^3$$

$$\text{sqrt}(Y_3) = 0.92690791 + 1.221648810e^x$$

### 11.3.4 FDD.3 系统

#### 1. 多维函数空间定义

多维函数空间由初等函数、初等函数组合、复合函数、复合函数组合、函数导数等组成。初等函数组合是初等函数之间运算组合;导数处理包括一阶差分、二阶差分、一阶差商、二阶差商等。多维函数空间的构造如下:

**定义 1** 设多维函数空间 $\Omega$:$\Omega=<P,V,C>$,其中:

$P=\{f_1, f_2,\cdots,f_m\}$是一个多元函数集,$f_i$ 是多元函数;

$V=\{v_1,v_2,\cdots,v_k\}$是一个有穷变元集;

$C=\{c_1,c_2,\cdots,c_k\}$是一个有穷常元集。

$P$ 函数集可以包括:

- 算术运算(如+,-,×,/等);
- 初等函数(如 1,$x^1$,$x^2$,$x^{1/3}$,sin,cos,exp,log 等一元函数);
- 导数函数。

#### 2. 多维函数空间性质

从以上定义可以看出,多维函数空间具有如下性质:

性质 1:在多维函数空间中,设 $E = V\cup C$,它满足条件:

(1) 对 $\forall e$,若 $e\in E$,则 $e\in\Omega$;

(2) 对 $\forall f,e_i$,若 $f \in P$, $e_i\in E$,则 $f(e_1,e_2,\cdots,e_n)\in\Omega$,$i=1,2,\cdots,n$,即函数作用于变元或常数仍然属于函数空间;

(3) 若 $p_1,p_2,\cdots,p_n\in\Omega$,则对 $\forall f \in P$, $f(p_1,p_2,\cdots,p_n)\in\Omega$,即函数作用于函数仍然属于函数空间。

性质 2:由于函数作用于变元或常数和函数作用于函数仍然是函数,故函数空间是封闭的。

对于在函数空间上的任意函数组合,仍然在函数空间中,这样为计算机对函数空间的处理提供了可以递归的前提。在函数空间中的函数集合可以组成解决问题的原型库。原型库一般包括初等函数、组合函数、复合函数,还包括差分计算、差商计算以及导数计算等。

#### 3. FDD.3 规则内容

系统中的知识采用产生式规则表示形式(if…then…),规则内容包括函数规则和控制规则,函数规则组成知识库,知识库不仅包括 FDD.1 系统规则,FDD.2 系统规则,还包括以下规则。

(1) 函数规则(FunRule)

对某一变量 $x$ 取函数空间中的一个函数 $f_i$ 后与另一变量 $y$ 的函数 $f_j$ 进行线性组合,得到函数公式后,代入 $x$ 和 $y$ 的值,取函数公式两边值的误差最小,则有函数公式:

$$C_1 f_i(x) + C_2 f_j(y) = C_3,\quad f_i,f_j \in P,\ C_1,C_2,C_3 \in C$$

(2) 函数嵌套规则

对函数规则嵌套或递归使用,将形成变量的任意组合。

(3) 误差规则(ErrRule)

- 误差最小规则:选择误差最小的公式进入下一次迭代;
- 误差收敛规则:保留误差减小的搜索方向,上一次迭代的误差大于目前的误差,则对于这一搜索方向予以保留。

(4) 终止规则(EndRule)

终止准则由两部分组成,一是强制终止,一是自然终止,强制终止通过对算法参数的设定,主要是通过对迭代次数的设定完成终止准则;自然终止有两种情况组成,一种是找到一组满足给定误差的公式,另一种情况是判断出误差增大时,则停止该路径的搜索。

(5) 多维函数扩展规则(MultiRule)

① 扩展到三维函数公式的启发式规则

设给定 $n$ 组不同的数据 $\{x_1^{(k)}, x_2^{(k)}, x_3^{(k)}\}, k=1,2,3,\cdots,n$,存在不同的函数 $f_1, f_2, f_3, f_4$ 以及常量 $C_1, C_2$, $B_1, B_2$,有如下函数关系:

- 如果在固定 $x_3$ 的情况下得出 $x_1$ 和 $x_2$ 的方程为

$$f_1(x_1) = C_1 f_2(x_2) + C_2 \tag{11.13}$$

在固定 $x_2$ 的情况下得出 $x_1$ 和 $x_3$ 的方程为

$$f_1(x_1) = B_1 f_3(x_3) + B_2 \tag{11.14}$$

从严格意义上讲,在式(11.13)中,常数 $C_1, C_2$ 是 $x_3$ 的函数;在式(11.14)中,常数 $B_1$,$B_2$ 是 $x_2$ 的函数。对于同一函数 $f_1(x_1)$应该有关于 $x_2$ 和 $x_3$ 的统一的公式,故对 $f_1(x_1)$而言,在式(11.14)中确定了 $x_1$ 与 $x_2$ 的关系,式(11.13)中确定了 $x_1$ 与 $x_3$ 的关系,合并式(11.13)与式(11.14),有如下启发式公式:

$$f_1(x_1) = C'_1 f_3(x_3) f_2(x_2) + C''_2 \tag{11.15}$$

$$f_1(x_1) = C'_1 f_2(x_2) + C'_2 f_3(x_3) + C''_3 \tag{11.16}$$

- 在固定 $x_2$ 的情况下得出 $x_1$ 和 $x_3$ 的方程为

$$f_3(x_1) = B'_1 f_4(x_3) + B'_2 \tag{11.17}$$

合并式(11.13)与式(11.17)则有如下多个启发式公式:

$$f_1(x_1)\theta f_3(x_1) = (c_1 f_2(x_2) + C_2)\theta(B'_1 f_4(x_3) + B'_2) \tag{11.18}$$

其中 $\theta$ 为+、-、*、/等操作。或者:

$$f_1(x_1) = g(x_1, x_2) + B'_1 f_4(x_3) + C_1 f_2(x_2) + C_3 \tag{11.19}$$

$g$ 函数的结构形式实质上是函数 $f_2$ 和 $f_3$ 的复合形式,由于 $f_1$ 和 $f_3$ 有系数项也有常数项,故 $f_1$ 和 $f_3$ 复合函数形式根据具体函数的不同有不同的合并方式,常见的是用一个公式的函数项去替换另外一个公式的系数和常数。

② 扩展到四维函数公式的启发式规则

设在三维数据的基础上增加一维数据 $x_4$,如果得到公式

$$f_2(x_2) = C_1 g(x_1, x_3) + C_2 \tag{11.20}$$

$$f_2(x_2) = C_3 f_4(x_4) + C_4 \tag{11.21}$$

则有如下启发式公式:

$$f_2(x_2) = C_1' g(x_1, x_3) f_4(x_4) + C_2' \quad (11.22)$$

$$f_2(x_2) = C_1 g(x_1, x_3) + C_3 f_4(x_4) + C_3'' \quad (11.23)$$

③ 多维函数的扩展

通过增加函数变量的方法可以实现对多维函数变量公式的发现。多维函数扩展规则给出了函数公式的具体框架表示形式,最后必须通过给定的数据对各个启发式公式进行检验,决定公式的取舍。首先,通过实际给出的数据应用最小二乘法计算上式中各个常量的值;其次通过给定的数据确定各个启发式公式的误差,最后进行选择,满足误差需求的公式即为所求公式。

#### 4. 三维函数公式的发现实例

(1) 试验数据

给定数据如表 11.11 所示。

**表 11.11　三维数据实例**

| $x_1$ | $x_2$ | $x_3$ | $x_1$ | $x_2$ | $x_3$ |
|---|---|---|---|---|---|
| 1.30 | 2.10 | 1.85 | 2.30 | 7.10 | 2.20 |
| 1.29 | 2.50 | 1.69 | 2.43 | 7.09 | 2.17 |
| 1.31 | 7.50 | 1.60 | 2.56 | 7.11 | 2.14 |
| 1.29 | 4.00 | 1.77 | 2.88 | 7.10 | 2.04 |
| 1.32 | 7.11 | 2.29 | … | … | … |

对于前 5 组数据,可以认为 $x_1$ 为恒定,应用二维函数公式发现算法,找出变量 $x_2$ 和 $x_3$ 的关系,得到 5 个公式,选择误差最小一个公式如下:

$$x_3^2 = 2.02 \times \cos(x_2) + 4.46 \quad 误差为:0.0016 \quad (11.24)$$

对于后 5 组数据,可以认为 $x_2$ 为恒定,应用二维函数公式发现算法,得到三个公式,选择误差最小的两个公式如下:

$$x_3^2 = 1.5 \times \sin(x_1) + 7.75 \quad 误差为:0.00026 \quad (11.25)$$

$$\log_{10}(x_3) = 0.07\sin(x_1) + 0.29 \quad 误差为:0.00015 \quad (11.26)$$

应用三维启发规则,将式(11.24)和式(11.25)合并,式(11.24)和式(11.26)合并,得到一系列公式,计算误差后得到满足误差要求的公式为

$$x_3^2 = 1.5 \times \sin(x_1) + 2.02 \times \cos(x_2) + 7.0 \quad (11.27)$$

该公式等式两端误差为:0.00041。

(2) 折射定律的发现

实验数据如表 11.12 所示(液体,温度为 20℃)。

设入射角为 $i$,折射角为 $\gamma$,入射线所在介质的折射率为 $n_1$,折射线所在介质的折射率为 $n_2$。因为光的可逆性,所以入射角和入射线的折射率与折射角和折射线的折射率两组数据可以互换,折射角 $\gamma$ 改为入射角 $i$,入射角 $i$ 变为折射角 $\gamma$,入射线和折射线所在位置的折射率也相应的调换。

表 11.12　不同介质间光线折射数据

| 物质 | 从空气中入射率 $n_1$($n_1$,$i$ 恒定) | | | 从空气射入玻璃($n_1$,$n_2$ 恒定) | |
|---|---|---|---|---|---|
| | 折射率 $n_2$ | 入射角 $i$ | 折射角 $\gamma$ | 入射角 $i$ | 折射角 $\gamma$ |
| 丙酮 | 1.3585 | 30 | 21.60 | 30 | 19.47 |
| 苯胺 | 1.5863 | 30 | 18.37 | 35 | 22.48 |
| 苯 | 1.5014 | 30 | 19.45 | 40 | 27.37 |
| 二硫化碳 | 1.6279 | 30 | 17.89 | 45 | 28.13 |
| 四氯化碳 | 1.4607 | 30 | 20.02 | 50 | 30.71 |
| 肉桂醛 | 1.6195 | 30 | 17.16 | 55 | 37.10 |
| 氯仿 | 1.4453 | 30 | 20.24 | 60 | 37.26 |
| 乙醇 | 1.3618 | 30 | 21.54 | | |

对于从空气中入射到各介质,固定 $n_1=1$ 和 $i=30$ 角后,应用二维函数公式发现算法,对不同物质得到折射率和折射角的公式:

$$\sin(\gamma) = 0.5/n_2 \tag{11.28}$$

反之,从介质中入射到空气时($n_1$ 变为 $n_2$,$i$ 角变为 $\gamma$ 角),固定 $n_2$ 和 $\gamma$ 角后,发现公式为

$$\sin(i) = 0.5/n_1 \tag{11.29}$$

现在固定空气和玻璃两种介质时($n_1=1$,$n_2=1.66$ 恒定),入射角 $i$ 和折射角 $\gamma$ 的关系,通过公式发现得:

$$\sin(i) = 1.5 \times \sin(\gamma) \tag{11.30}$$

式(11.28)和式(11.29)两个公式从空气中入射不同物质的数据中生成,式(11.30)为从空气中入射玻璃的一组数据中生成。式(11.29)和式(11.30)应用三维扩展规则得:

$$\sin(i) = C_1 \times \sin(\gamma)/n_1 + C_2,\text{即}$$

$$\sin(\gamma) = C_1^* \times \sin(i) \times n_1 + C_2^* \tag{11.31}$$

对式(11.28)和式(11.31)利用四维扩展规则进行合并,得:

$$\sin(\gamma) = C_1'' \times \sin(i) \times (n_1/n_2) + C_2'' \tag{11.32}$$

用已知的数据确定系数,得 $C_1''=1$,$C_2''=0$,即得 Snell 折射定律:

$$\sin(i) \times n_1 = \sin(\gamma) \times n_2 \tag{11.33}$$

**5. FDD.1、FDD.2 和 FDD.3 的比较分析**

FDD.2 是通过引入导数规则对 FDD.1 算法得规则进行扩充,同时修改算法流程,使得算法运行更加合理,扩大了发现公式的宽度和广度。FDD.3 算法引入多维函数处理规则后对 FDD.2 算法进行了扩充,同时通过嵌套 FDD.2 算法流程,实现了三维以上公式发现算法 FDD.3。把这三个进行比较分析,如表 11.13 所示。

表 11.13　FDD.1、FDD.2 和 FDD.3 的比较分析

| 比较方面 | FDD.1 | FDD.2 | FDD.3 |
|---|---|---|---|
| 时间复杂度 | $O(8nm)$ | $O(2n^2m)$ | $O(C_d^2 2n^2 m)$ |
| 流程循环 | 函数作用于一个变量 | 不同的函数作用于两个变量 | |
| 剪枝条件 | 误差最小原则 | 误差最小原则<br>误差收敛原则 | 误差最小原则<br>误差收敛原则 |
| 发现公式范围 | 初等函数、复合函数及其组合 | 在 FDD.1 基础上增加导数以及和导数相关的处理 | 在二维 FDD 基础上增加：<br>三维扩展规则<br>多维扩展规则 |

说明：$n$ 为函数个数，$m$ 为搜索树的深度，$d$ 为维数。

在进行算法的时间复杂度分析时，由于搜索树的剪枝根据具体情况的不同而不同，所以假设在没有剪枝的情况下分析各个算法的时间复杂度。由于算法流程的不同，在发现同样形式的公式情况下，FDD.1 和 FDD.2，FDD.3 搜索树的深度不同，FDD.1 算法搜索树深度是 FDD.2，FDD.3 算法的两倍。

在 FDD.1 算法中，每个函数对两个变量分别作用的时间复杂度 $O(2n)$，选择两个误差小的进入下面的分支，并且树的深度是 $2m$，则时间复杂度为 $O(8nm)$。

在 FDD.2 算法中，两个函数同时作用于两个变量时间复杂度为 $O(nn)$，选择误差小的和误差收敛的进入下一个循环，则时间复杂度为 $O(2nnm)$。在 FDD.3 算法中，设函数的维数为 $d$，则任取其中的两个变量的组合为 $C_d^2$ 个，所以整个算法的时间复杂度为 $O(C_d^2 2n^2 m)$。FDD.3 算法的发现公式的广度是以牺牲时间为代价的。

BACON 系统采用“项＝常数”的形式描述公式形式，而 FDD 采用“项＝初等函数或初等函数的复合形式”，并且引入导数规则等，和 BACON 相比发现公式的范围和复杂度都有很大提高。

# 习　题　11

1. 数据拟合的基本思想是什么？有哪些优点和缺点？
2. 从 BACON 系统的实例看，公式发现与数据拟合有什么不同？
3. BACON 系统的简练算子有哪些？
4. BACON 系统是如何完成开普勒第三定律的发现的？
5. BACON 系统是如何发现理想气体定律的？
6. BACON 系统的启发式是什么？
7. 科学定律运用曲线拟合能发现吗？
8. FDD 系统的思想是什么？
9. FDD.1 系统的启发式函数是什么？
10. FDD.1 系统结构图的基本思想是什么？
11. FDD.1 系统中函数原型有哪些？

12. FDD.1 系统中的知识有哪些?

13. FDD.1 系统完成开普勒第三定律的现的过程是什么? 它与 BACON 系统的发现过程有什么不同?

14. FDD.2 发现导数公式的启发式是什么?

15. FDD.3 发现多维函数公式的启发式是什么?

16. FDD 系统与 BACON 系统有什么不同?

# 第12章 知识挖掘

## 12.1 变换规则的知识挖掘

本节讨论一种新的规则知识，即含变换的规则知识，称为变换规则知识，这是一种适应变化环境的新知识，也是元知识的一种新的表示形式。变换规则知识的挖掘是在数据挖掘获得的规则知识的基础上，加上规则中前提或结论的变换，获得变换规则知识。

### 12.1.1 适应变化环境的变换和变换规则

#### 12.1.1.1 数学变换与可拓变换

**1. 数学变换**

(1) 数学中的函数是一种变换，如 $y=f(x)$表示把 $x$ 值经过函数计算变换成 $y$ 值。

(2) 数学中的变量求值也是一种变换，如方程 $f(x)=0$ 的求解，实质上是对变量 $x$ 通过方程的求解，得到 $x$ 的具体值，即变量 $x$“从未知到已知”的变换，可称“求值变换”。

(3) 计算机中的过程是向量变换，如过程 $F(X,Y)$表示把输入向量 $X(x_1,x_2,\cdots,x_n)$值经过过程计算变换成输出向量 $Y(y_1,y_2,\cdots,y_m)$值。

(4) 数学中的坐标变换，如坐标的平移和旋转，把曲线(曲面)方程的一般形式变换成标准形式。使不清晰的方程变换成清晰的椭圆、抛物线(面)、双曲线(面)等标准方程。

(5) 数学中的积分变换，如拉普拉斯变换把不能求解的微分方程变换成可求解的代数方程。在代数方程求出解后，再通过拉普拉斯逆变换，把代数方程的解变换成微分方程的解。

以上数学变换均把未知的变量、不能求解的方程变换成已知的量值、能求解的方程，体现了定量变化的特点。

**2. 可拓变换**

在可拓学中，利用可拓变换来解决矛盾问题。

**定义 1** 可拓变换定义为对对象(物元、事元、关系元、准则、论域)的变换，即：

$$Tu = v \tag{12.1}$$

可拓变换 $T$ 将对象 $u$ 变为对象 $v$。可拓变换包括置换变换、增删变换、扩缩变换、分解变换、复制变换等。

**3. 变换的逻辑表示**

数学变换或可拓变换在此统称为变换。

变换将对象 $u$ 变为对象 $v$，实际上完成了 $u$ 自身变为$\sim u$，并使 $v$ 成为真。这样，变换可

以用形式逻辑表示。

**定义 2** 变换的形式逻辑表示为

$$Tu = v \leftrightarrow \sim u \wedge v \tag{12.2}$$

**4. 变换的宏观抽象作用**

(1) 数学和计算机中的变换概括了函数、求值、过程的概念，是隐含了具体的计算，抽象为一个宏观变换。实质上是"从定量到定性"的抽象。

(2) 专家系统的目标包含了多个取值，对不同问题目标取值是不同的，对一个实际问题的目标取值是通过知识推理来获得的。它实质上是一个目标求值的变换，起到了宏观的"从定性到定性" 抽象的作用。

可见，"变换"可以是简单变换，即把一个具体的对象变换成另一个具体的对象。"变换"可以是复杂的宏观的变换，即把一个目标变换成另一个目标。目标既可以是一系列定量计算过程的抽象，也可以是多次定性推理过程的抽象。

### 12.1.1.2 变换规则

变换可能由某个条件(原因)产生或者变换会引起某个结果。本书作者在变换的基础上提出了变换产生式，即变换规则概念。

**1. 变换 *T* 由某一条件或原因所引起**

$$\text{Condition} \rightarrow Tu = v \tag{12.3}$$

(1) 条件 Condition 可能是某一事实 $F=f$，具体表示为

$$F = f \rightarrow Tu = v \tag{12.4}$$

(2) 条件 Condition 可能是另一个变换 $Ta=b$，具体表示为

$$T_a a = b \rightarrow T_u u = v \tag{12.5}$$

注意：为区分不同的变换，在变换的下角加以标注，即 $T_a$、$T_u$。

(3) 条件 Condition 可能是一个算子 $A$ 求出变量 $X$ 的值，表示为

$$A(x) = b \rightarrow Tu = v \tag{12.6}$$

**2. 变换 *T* 产生一个结果**

$$Ta = b \rightarrow \text{result} \tag{12.7}$$

结果 result 同样可能是一个事实，或者是另一个变换。

**3. 变换规则定义**

**定义 3** 包含变换的规则，即与变换有关的具有产生式关系的规则式，统称为变换规则，或称变换产生式。

变换规则是一种新的知识表示形式。这种新的知识，用于解决矛盾问题时，称为可拓知识。在式(12.3)～式(12.7)中，式(12.5)是典型的变换规则的代表形式。在可拓学中，式(12.5)中结论的可拓变换称为前提可拓变换的传导变换，变换规则知识称为可拓知识。

**4. 变换规则知识与规则知识的对比**

(1) 规则知识

在人工智能中一般知识表示成规则形式,即规则知识,表示为

$$P \rightarrow Q$$

其中 $P$ 与 $Q$ 均为事实(变量的取值)它表示事实 $P$ 是事实 $Q$ 的原因,事实 $Q$ 是事实 $P$ 的结果。知识只体现了 $P$ 与 $Q$ 两个事实间的静态关系。

(2) 变换规则知识

变换规则知识中,规则的前项或者后项中包括了变换,而变换将一对象变换为另一个对象,体现了变化的特点。

公式(12.5)表示变换 $T_a$ 把 $a$ 变换 $b$,引起了另一个变换 $T_u$ 把 $u$ 变成 $V$,这种变换规则知识完全体现了变化的情况,因此,变换规则知识是适应变化的知识,相对而言,人工智能的知识是静态知识。也可以说变换规则知识是知识的推广,是一种更有价值的知识。

## 12.1.2 变换规则的知识挖掘的理论基础

数据挖掘是利用算法获取规则知识(条件→结论)。我们在数据挖掘获取知识的基础上,若规则的条件和结论都存在变换,将获得变换规则知识:

$$T_{条件} \rightarrow T_{结论}$$

把这种挖掘变换规则知识称为新型的变换规则的知识挖掘,即在规则知识的基础上挖掘变换规则知识。它不同于数据挖掘是在数据的基础上挖掘知识。

### 12.1.2.1 变换规则的知识挖掘定理

**定理 1** 对于两类规则

$$A \rightarrow P \tag{12.8}$$

$$B \rightarrow N \tag{12.9}$$

一般情况 $A = \wedge a_i, B = \wedge b_i$。

若存在条件的变换 $T_B$

$$T_B(B) = A \tag{12.10}$$

并存在结论的变换 $T_N$

$$T_N(N) = P \tag{12.11}$$

则成立变换规则知识

$$T_B(B) = A \rightarrow T_N(N) = P \tag{12.12}$$

即

$$\text{if } T_B(B) = A \text{ then } T_N(N) = P \tag{12.13}$$

**证明:**

(1) 定理的已知条件表示成命题逻辑公式,并化为子句型

a. $A \rightarrow P \leftrightarrow \neg A \vee P$;

b. $B \rightarrow N \leftrightarrow \neg B \vee N$;

c. $T_B(B) = A \leftrightarrow \neg B \wedge A \leftrightarrow \neg B, A$;

d. $T_N(N) = P \leftrightarrow \neg N \wedge P \leftrightarrow \neg N, P$。

(2) 对定理的结论取非后化成子句型

$\neg(T_B(B) = A \rightarrow T_N(N) = P) \leftrightarrow \neg[(\neg B \wedge A) \rightarrow (\neg N \wedge P)] \leftrightarrow$

$\neg[\neg((\neg B) \wedge A) \vee (\neg N \wedge P)] \leftrightarrow \neg[(B \vee \neg A) \vee (\neg N \wedge P)] \leftrightarrow$

$\neg(B \vee \neg A) \wedge \neg(\neg N \wedge P) \leftrightarrow \neg B \wedge A \wedge (N \vee \neg P) \leftrightarrow \neg B, A, N \vee \neg P$。

(3) 对全部子句集进行归结

a. 全部子句集为

$$\neg A \vee P, \neg B \vee N, \neg B, A, \neg N, P, N \vee \neg P$$

b. 归结过程

子句 $\neg A \vee P$ 与子句 $A$ 归结为 $P$,它与子句 $N \vee \neg P$ 归结为 $N$,再和子句 $\neg N$ 归结为空子句,产生矛盾,故证明定理正确。

**定理 2** 对于两条同类规则

$$A \rightarrow P \tag{12.14}$$

$$C \wedge B \rightarrow P \tag{12.15}$$

若存在可拓变换 $T_B$

$$T_B(B) = A \tag{12.16}$$

则成立:可拓变换规则知识

$$T_B(B) = A \rightarrow P \tag{12.17}$$

即

$$\text{if } T_B(B) = A \text{ then } P \tag{12.18}$$

该定理同样可用归结原理证明,此处省略。

#### 12.1.2.2 变换规则的知识挖掘过程

从变换规则的知识挖掘定理中,可以概括变换规则的知识挖掘过程为:

Step 1 对分类问题利用数据挖掘方法获得分类规则,即获得式(12.8)和式(12.9)的规则知识。

Step 2 确定规则的前提中存在的变换以及结论中存在的变换,即找出满足式(12.10)和式(12.11)的变换。

Step 3 利用定理 1 和定理 2 获得变换规则的知识式(12.12)或式(12.17)。

#### 12.1.2.3 变换规则的知识挖掘实例

在本书 7.2.2 节中,对表 7.1 气候训练集,利用 ID3 方法得到决策树知识(见图 7.4),将它转换为规则知识(树中从根结点到叶结点的每一条路径构成一条知识)。

(1) 数据挖掘获取的规则知识

if 天气=晴 and 湿度=正常 then 类别=$P$

if 天气=多云 then 类别=$P$

if 天气=雨 and 风=无风 then 类别=$P$

if 天气＝晴　and 湿度＝高　then 类别＝$N$

if 天气＝雨　and 风＝有风　then 类别＝$N$

(2) 存在的变换

① 条件变换

$$T_1(天气 = 晴) = (天气 = 多云)$$
$$T_2(天气 = 晴) = (天气 = 雨)$$
$$T_3(天气 = 雨) = (天气 = 多云)$$
$$T_4(天气 = 多云) = (天气 = 晴)$$
$$T_5(天气 = 雨) = (天气 = 晴)$$
$$T_6(天气 = 多云) = (天气 = 雨)$$
$$T_7(湿度 = 高) = (湿度 = 正常)$$
$$T_8(湿度 = 正常) = (湿度 = 高)$$
$$T_9(风 = 无风) = (风 = 有风)$$
$$T_{10}(风 = 有风) = (风 = 无风)$$

② 结论变换

$$T(N) = P$$
$$T(P) = N$$

(3) 利用变换规则的知识挖掘的定理 1 和定理 2,可以得到变换规则知识

① 类别发生变化的知识

$(天气 = 晴)$ and $(T_7(湿度 = 高) = (湿度 = 正常)) \to T(N) = P$

$(湿度 = 高)$ and $(T_1(天气 = 晴) = (天气 = 多云)) \to T(N) = P$

$(天气 = 雨)$ and $(T_{10}(风 = 有风) = (风 = 无风)) \to T(N) = P$

$(风 = 有风)$ and $(T_3(天气 = 雨) = (天气 = 多云)) \to T(N) = P$

$(天气 = 晴)$ and $(T_8(湿度 = 正常) = (湿度 = 高)) \to T(P) = N$

$(天气 = 雨)$ and $(T_9(风 = 无风) = (风 = 有风)) \to T(P) = N$

② 类别不发生变化的知识

$(湿度 = 正常)$ and $(T_1(天气 = 晴) = (天气 = 多云)) \to 类别 = P$

$(风 = 无风)$ and $(T_3(天气 = 雨) = (天气 = 多云)) \to 类别 = P$

$(风 = 无风)$ and $(T_6(天气 = 多云) = (天气 = 雨)) \to 类别 = P$

$(湿度 = 正常)$ and $(T_4(天气 = 多云) = (天气 = 晴)) \to 类别 = P$

这些变换规则知识告诉人们,在天气变化时,类别会不会发生变化。这种适合变化环境的变换知识,比静态知识有时更有用。

### 12.1.3　变换规则的知识推理

在智能科学中,知识推理采用了形式逻辑中的假言推理。变换规则知识的推理是对变换规则知识的假言推理。

#### 12.1.3.1 变换规则的知识推理式

**定义 4** 变换规则知识的假言推理表示为

$$(T_u u = u') \land [(T_u u = u') \to (T_v v = v')] \vdash (T_v v = v') \tag{12.19}$$

变换规则知识的推理是在知识推理的基础上扩展为对变换规则知识的推理。下面证明变换规则知识推理式(12.19)是正确的。

**证明：**

(1) 将式(12.19)中推理(⊢)的左部写成等价的命题逻辑公式

$$(\neg u \land u') \land [(\neg u \land u') \to (\neg v \land v')]$$

(2) 上式化为子句型

$$(\neg u \land u') \land [(\neg u \land u') \to (\neg v \land v')] \leftrightarrow$$
$$(\neg u \land u') \land [\neg (\neg u \land u') \lor (\neg v \land v')] \leftrightarrow$$
$$(\neg u \land u') \land [(u \lor \neg u') \lor (\neg v \land v')] \leftrightarrow$$
$$(\neg u \land u') \land [(u \lor \neg u' \lor \neg v) \land (u \lor \neg u' \lor v')] \leftrightarrow$$
$$(\neg u \land u') \land (u \lor \neg u' \lor \neg v) \land (u \lor \neg u' \lor v') \leftrightarrow$$
$$\neg u,\ u',\ (u \lor \neg u' \lor \neg v),\ (u \lor \neg u' \lor v')$$

(3) 将推理(⊢)的右部取非后，化为子句型

$$\neg (Tv = v') \leftrightarrow \neg(\neg v \land v') \leftrightarrow v \lor \neg v'$$

(4) 归结过程

子句 $v \lor \neg v'$ 与子句 $(u \lor \neg u' \lor \neg v)$ 归结为 $\neg v' \lor u \lor \neg u'$，它与子句 $\neg u$ 归结为 $\neg v' \lor \neg u'$，与 $u'$ 归结为 $\neg v'$，再与子句 $(u \lor \neg u' \lor v')$ 归结为 $u \lor \neg u'$，与 $\neg u$ 归结为 $\neg u'$，再与 $u'$ 归结为空子句。产生矛盾，证明可拓推理式(12.19)是正确的。

变换规则知识只表明存在对象变化的可能性。变换规则知识的推理表明实际对象变化的发生。在式(12.19)中，变换规则知识 $(T_u \to T_v)$ 只表明对 $u$ 的变换 $T_u$ 会引起对 $v$ 的变换 $T_v$。在推理式中现已发生变换 $T_u$，按推理式的推理必然出现变换 $T_v$。

#### 12.1.3.2 变换规则的知识挖掘实例

在“脑血栓”与“脑出血”两类疾病的数据库中进行数据挖掘和变换规则的知识挖掘。

**1. 在数据库中通过数据挖掘获取规则知识**

从“脑出血”和“脑血栓”两种疾病的大量实例数据库中，通过数据挖掘的遗传算法可以获取两种疾病独立诊断的规则知识。获得的主要 7 条规则(具体数据挖掘过程从略)：

(1) (高血压＝有)∧(瞳孔不等大＝是)∧(膝腱反射＝不活跃)→脑出血；

(2) (瞳孔不等大＝是)∧(语言障碍＝是)→脑出血；

(3) (高血压＝有)∧(起病方式＝快)∧(意识障碍＝深度)→脑出血；

(4) (高血压＝有)∧(病情发展＝快)→脑出血；

(5) (高血压＝有)∧(动脉硬化＝有)∧(起病方式＝慢)→脑血栓；

(6) (动脉硬化＝有)∧(病情发展＝慢)→脑血栓；

(7)(动脉硬化＝有)∧(意识障碍＝无)→脑血栓。

**2. 确定存在的条件变换和结论变换**

在医疗中病人存在的条件变换有：

$$T_{条件}(起病方式慢) = 起病方式快$$

$$T_{条件}(无意识障碍) = 深度意识障碍$$

也存在结论变换：

$$T_{结论}(脑血栓) = 脑出血$$

**3. 利用变换规则的知识挖掘理论获取变换规则知识**

根据定理 1 得到变换规则知识为

$$T(有动脉硬化 \wedge 起病方式慢 \wedge 无意识障碍) = 起病方式快 \wedge 有深度意识障碍 \rightarrow T(脑血栓) = 脑出血 \quad (12.20)$$

还可以得出其他的变换规则知识。

**4. 变换规则知识的推理**

变换规则知识中的前提一旦在现实中出现，就可以利用变换规则知识的推理判断变换规则知识中结论的出现。当发现某病人由"起病方式慢"变成"起病方式快"，同时"无意识障碍"变成"有深度意识障碍"，即变换规则知识式(12.20)的前提已经出现，利用变换规则知识的推理式(12.19)就可以判断变换规则知识式(12.20)的结论已经出现，即应该诊断该病人已经由"脑血栓"变成了"脑出血"。治疗方式就应改由"脑血栓"的治疗方法变成治疗"脑出血"的方法。

两种疾病的治疗方法是完全相反的，"脑血栓"的治疗方法是通血管，使血流通畅。而"脑出血"的治疗方法是堵血管，不让血流外溢。当"脑血栓"已变成了"脑出血"后，若仍然用"脑血栓"的治疗方法治疗"脑出血"，即继续通血管，这样只可能造成更大范围的脑出血，将会加重"脑出血"症状，甚至于导致死亡。这条变化知识对医生来讲是极其重要的。

可见，挖掘具有变化特点的变换规则的知识挖掘比挖掘静态规则知识的数据挖掘更有意义。

## 12.1.4 变换规则链的知识挖掘

### 12.1.4.1 基于集合的变换规则知识

在集合论中有集合蕴含关系，定义如下：

**定义 5** 若集合 $P$ 和 $Q$ 存在关系 $P \subseteq Q$，则成立蕴含关系

$$P \rightarrow Q \quad (12.21)$$

即集合 $P$ 中的元素 $x$ 一定属于集合 $Q$。由此定义可以得到如下定理：

**定理 3** (基于集合的变换规则)对于变换 $T_a a = b$ 和变换 $T_e e = f$，若存在集合关系 $a \subseteq e, b \subseteq f$，则存在变换规则知识：

$$T_a a = b \rightarrow T_e e = f \tag{12.22}$$

简写为：$T_a \rightarrow T_e$，并称变换 $T_a$ 与 $T_e$，是同类变换，即两个变换前的对象$\{a,e\}$与两个变换后的对象$\{b,f\}$均在各同类集合中。

**证明：**

(1) 由于 $a \subseteq e$，由定义 5 可知，存在蕴含关系：

$$a \rightarrow e \tag{12.23}$$

(2) 由于 $b \subseteq f$，同样存在蕴含关系：

$$b \rightarrow f \tag{12.24}$$

根据定理 1 可知，对于式(12.23)和式(12.24)，存在可拓变换 $T_a a = b$ 和 $T_e e = f$，则存在变换规则知识：

$$T_a a = b \rightarrow T_e e = f \tag{12.25}$$

#### 12.1.4.2　基于本体的变换规则知识链

本体(ontology)是目前研究最多的知识表示形式，本体是共享概念的规范化说明，本体在概念分类层次的基础上，加入了关系、公理、规则来表示概念之间的关系。

**定义 6**　(本体)本体由概念、关系、函数、公理和实例等五类基本元素构成，表示为如下形式：

$$O = [C、R、F、A、I] \tag{12.26}$$

其中，$C$ 为概念，$R$ 为关系，$F$ 为函数，$A$ 为公理，$I$ 为实例。关系 $R$ 有 4 种：subclass-of(或kind-of，子类)、part-of(部分)、instance-of(实例)和 attribute-of(属性)。

本体概念树的层次关系主要是 subclass-of 关系，即树的下层概念是上层概念的子集，如图 12.1 所示。

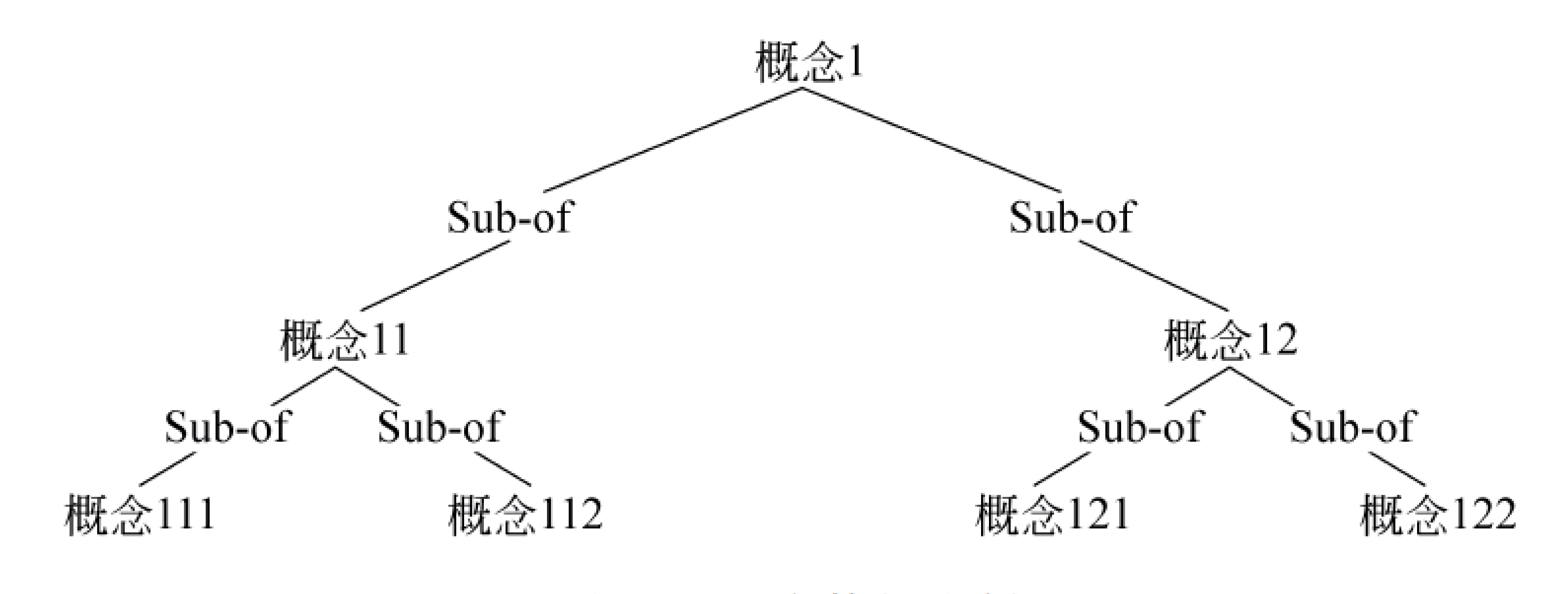



图 12.1　本体概念树

概念 11 的是概念 1 的子集，而概念 111 的是概念 11 的子集等等。

根据本体概念树的特点和定理 3，可以得到如下定理：

**定理 4**　本体概念层次关系中，下层概念的变换 $T_d$ 与上层概念的同类变换 $T_u$，存在变换规则：

$$T_d \rightarrow T_u \tag{12.27}$$

**证明：**

本体概念层次关系中，下层概念集合 $S_d$ 与上层概念集合 $S_u$ 存在蕴含关系：$S_d \subseteq S_u$。

根据定理 3 可知，下层概念集合 $S_d$ 中的变换 $T_d$ 与上层概念集合 $S_u$ 中的同类变换 $T_u$

存在变换规则的蕴含关系,即变换规则:

$$T_d \rightarrow T_u \tag{12.28}$$

**定理 5** (基于本体的变换规则链)在本体概念树中,叶结点中的变换 $T_0$ 与各级上层结点中的同类变换 $T_i$ 之间形成了变换规则链,即:

$$T_0 \rightarrow T_1 \rightarrow T_2 \rightarrow \cdots \rightarrow T_{root} \tag{12.29}$$

**证明**:由定理 3 可知,本体概念树的上下两层的同类变换都存在蕴含关系(变换规则知识)。由本体概念树叶结点开始,逐层向上到本体概念树的根结点,将同类变换连接起来,就形成公式(12.29)的变换规则链。

#### 12.1.4.3 多维层次数据中原因分析的变换规则链获取实例

在我国航空公司数据仓库中,对发现的问题进行原因分析,从中获取变换规则链。数据仓库中的多维数据中含层次粒度的大量数据,对发现的问题进行原因分析主要是进行多维数据的钻取操作。在每一次钻取中进行一次变换,获得出现问题原因的深层数据。数据仓库中的多维层次数据集合是符合本体概念树的层次关系。

我国航空公司的数据仓库的多维分析中发现了"北京到西南地区总周转量相对去年出现负增长"的问题,该问题的本体概念树如图 12.2 所示。

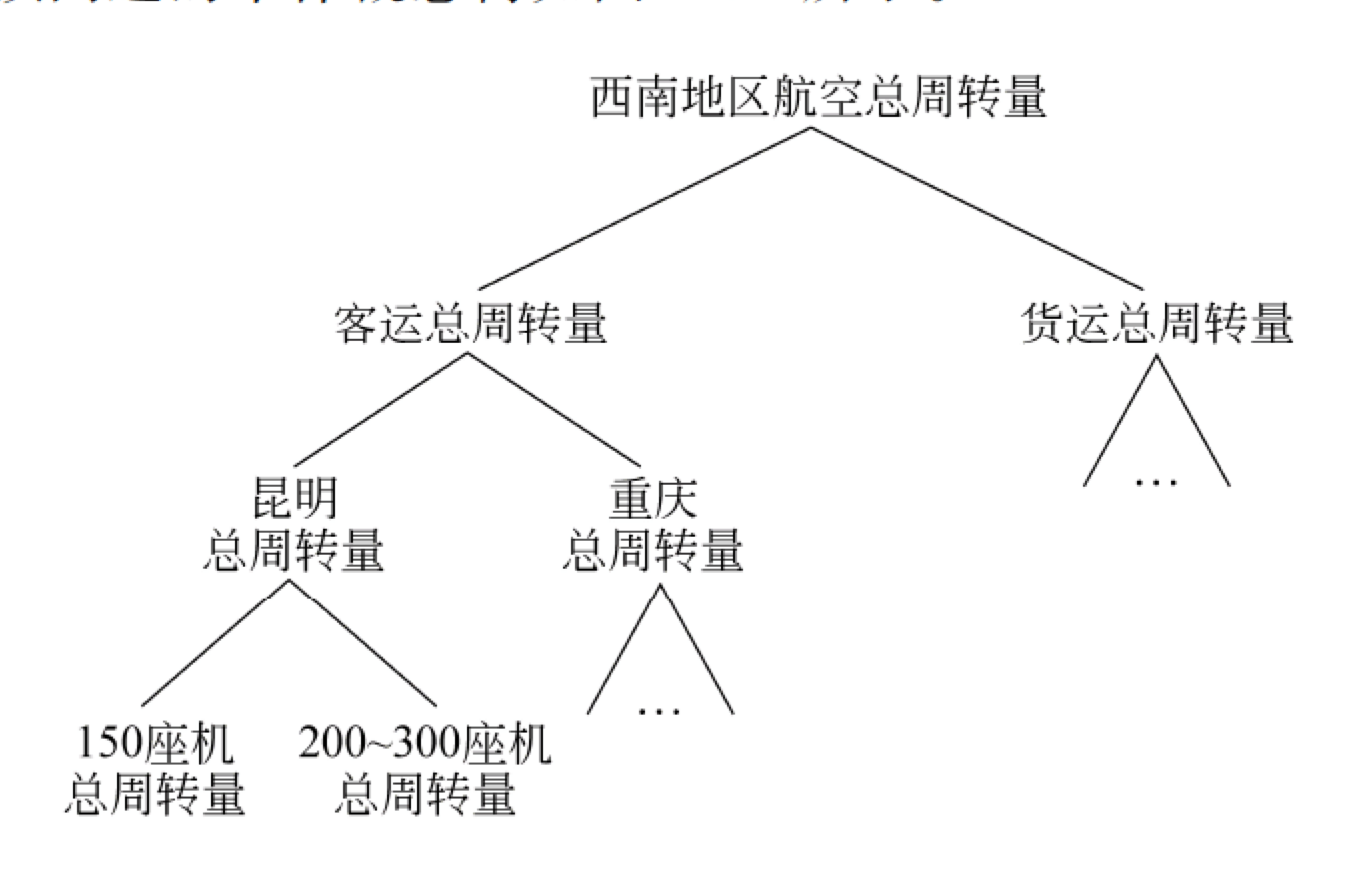



图 12.2 西南地区航空总周转量的本体概念树

该问题在本体树的根结点航空总周转量上的减变换表示为

$T_{西南总量}$(今年总周转量 — 去年总周转量)=— 19.9(负增长)

通过下钻到本体树下层,空运总周转量结点上的减变换为

$T_{西南客运}$(今年客运总周转量 — 去年客运总周转量)=— 19.4(负增长)

再下钻到昆明客运总周转量结点上的减变换为

$T_{昆明客运}$(今年总周转量 — 去年总周转量)=— 16.5(负增长)

再下钻到昆明座机为 150 座级与 200～300 座级机型的总周转量两个结点上的减变换分别为:

$T_{150座机}$(今年总周转量 — 去年总周转量)=— 6.83(负增长)

$T_{200\sim300座机}$(今年总周转量 — 去年占用转量)=— 8.9(负增长)

根据定理 5,可得到变换规则链为

$$T_{150座机} \wedge T_{200\sim300座机} \rightarrow T_{昆明客运} \rightarrow T_{西南客运} \rightarrow T_{西南总量} \tag{12.30}$$

该变换规则链说明：出现西南地区总周转量相对去年出现较大负增长，原因之一主要是昆明地区150座机和200～300座机型，相对去年出现较大负增长造成的。而该变换规则链的获得是从问题结论的减变换，($T_{西南总量}$)出现负增长，通过多维数据钻取，逆向找它的前提减变换，再向下钻取，一直到最底层(叶结点)中的减变换，即($T_{150座机}$及$T_{200\sim300座机}$)出现较大的负增长，该叶结点的减变换才是本体根结点问题的根本原因。

在向下钻取过程中，有时也能发现新问题，如在搜索货运总周转量时，发现东南地区出现了一个大负增长，这是除西南地区出现负增长外新发现的问题，可以在寻找西南地区航空总周转量的根本原因之后，再去寻找东南地区出现货运总周转量出现负增长的原因。

除了寻找负增长以外，还可以寻找正增长的原因，即从正、负两个方面寻找问题产生的原因，这样可以得到更大的决策支持。

寻找问题原因让计算机自动完成，必须建立多维层次数据的本体概念树，并在树中进行深度优先搜索，来发现问题并找到所有原因。

#### 12.1.4.3　小结

数据挖掘是从数据中挖掘知识，变换规则的知识挖掘是在规则知识的基础上挖掘变换规则知识。规则知识是静态的，而变换规则是变化的知识。变换规则定理帮助我们从规则知识及相关的变换中获取变换规则知识。基于本体的变换规则链定理帮助我们在数据仓库中多维层次数据中获取变换规则链。

目前，对数据仓库的问题的分析基本上是在人的指导下，对多维层次数据进行钻取操作，找到问题发生的原因。若在多维层次数据中建立本体概念树，就可以让计算机沿着本体概念树进行深度优先搜索，既可以发现问题，又能自动找到各问题的所有原因。这项工作是很有意义的。

### 12.1.5　适应变化环境的变换规则元知识

元知识是知识的知识，是对一般知识的描述、概括、处理、使用的知识。我们在此提出用变换规则作为元知识的一种新表示形式。变换规则是以变换为基础，是变换的产生式，它具有变化的特点，适应变化的环境。这种新的元知识表示称为变换规则元知识。

#### 12.1.5.1　神经网络的变换规则元知识

神经网络模型是将人脑神经元组织结构用数学模型进行形式化表示。它学会实际样本需要利用两个原理性的计算公式：①神经网络模型的运行机制：由输入结点值，经过MP模型计算公式，计算出输出结点的值；②利用输出结点值的误差，修正网络权值和阈值的计算公式(在本书9.3.2节中已说明)。此处利用变换规则作为元知识的表示形式进行神经网络计算的概括。

**1. 建立判别函数**

给定一个小数$\varepsilon$，利用误差函数

$$\delta_i = \sum_j | O_j^i - t_j^i |$$

其中，$t_j^i$ 是给定样本 $i$ 的输出结点 $y_j^i$ 的实际值，$O_j^i$ 是输出结点 $y_j^i$ 的计算输出值。

建立判别函数值为：$K_i = \varepsilon - \delta_i$，$K_i < 0$ 表示神经网络未学会样本，$K_i \geqslant 0$ 表示神经网络学会样本。

**2. 确定解决问题的变换**

解决该问题需要引入 5 个变换，分别是：

（1）输入结点到输出结点的变换 $T_{IO}$

利用 MP 模型，将输入结点值：$I = (x_1, x_2, \cdots, x_n)$，按神经网络公式计算，得出输出结点值：$O = (O_1, O_2, \cdots, O_m)$，其变换为

$$T_{IO}(I) = O$$

该变换的计算公式为

$$O_j = f(\sum_i w_{ij} x_i - \theta_j)$$

（2）输出结点的减变换 $T_-$

将样本输出结点的计算值与实际值进行相减，得到误差，即：

$$T_-(O) = \sum_j | o_j^i - t_j^i | = \delta_i$$

（3）网络权值的变换 $T_w$：

$$T_w(w_{ij}^{(k)}) = w_{ij}^{(k+1)}$$

该变换的计算公式为

$$w_{ij}^{(k+1)} = w_{ij}^{(k)} + \eta \delta_j x_i$$

（4）阈值的变换 $T_\theta$：

$$T_\theta(\theta_j^{(k)}) = \theta_j^{(k+1)}$$

该变换的计算公式为

$$\theta_j^{(k+1)} = \theta_j^{(k)} + \eta \delta_j$$

（5）判别函数值的变换 $T_k$：

$$T_K(K_i) = K_{i+1}$$

**3. 神经网络学会样本的变换规则元知识表示为**

$$T_{IO} \wedge T_- \wedge T_W \wedge T_\theta \rightarrow T_K \tag{12.31}$$

该变换规则表示，经过 4 个变换 $T_{IO}$、$T_-$、$T_w$、$T_\theta$ 将引起判别函数值的变换 $T_k$，使判别函数的值增加。该变换规则元知识高度概括了是神经网络学会样本的关键和解决过程。

**4. 算法**

（1）首先要给定神经网络上的网络初始权值和阈值（随机数），即 $w_{ij} = w_{ij}^{(0)}$，$\theta_i = \theta_i^{(0)}$。

（2）反复进行变换规则知识的计算，直到判别函数 $K_i \geqslant 0$ 为止。

（3）输出网络权值的结果：$w_{ij}^* = w_{ij}^{(n)}$，$\theta_i^* = \theta_i^{(n)}$。

### 12.1.5.2 知识发现的变换规则元知识

在知识发现中，属性约简和数据挖掘是两个重要步骤，这一节利用粗糙集理论，用变换规则元知识来高度概括这两个步骤的本质。

**1. 属性约简的变换规则元知识**

属性约简问题是在数据库中保持分类效果不变的情况下，删除多余的属性。它的基础理论主要是粗糙集理论和信息论。按粗糙集理论，需要对数据库中的每个条件属性计算其重要度 SGF，为此引入计算重要度算子 $A_{SGF}$。对条件属性集 $C$ 中的任一属性 $c_i$ 相对决策属性 $D$，计算其重要度 $A_{SGF}(c_i)$。

$A_{SGF}(c_i)$算子计算过程

(1) 计算条件属性集$(C-\{c_i\})$的等价集；

(2) 计算决策属性的 $D$ 等价集；

(3) 计算正域 $\mathrm{Pos}(C-\{c_i\},D)$；

(4) 计算依赖度 $\gamma(C-\{c_i\},D)$；

(5) 计算 $C_i$ 的重要度 $\mathrm{SGF}(C-\{c_i\},D)$。

在粗糙集属性约简中，若 $\mathrm{SGF}(C-\{c_i\},D)=0$，即 $A_{SGF}(c_i)=0$。表示属性 $c_i$ 关于 $D$ 是可省的，即可以对属性 $c_i$ 进行约简，用下式表示属性约简变换：

$$T_{reduc}(C) = (C - \{c_i\})$$

该约简变换 $T_{reduc}$ 是在算子 $A_{SGF}(c_i)$ 计算出 $A_{SGF}(c_i)=0$ 时才进行的变换。

算子 $A_{SGF}$ 与约简变换 $T_{reduc}$ 之间的因果关系可以表示为变换规则元知识：

$$A_{SGF}(c_i) = 0 \to T_{reduc}(C) = (C - \{c_i\}) \tag{12.32}$$

该元知识表示为：若算子 $A_{SGF}$ 对 $c_i$ 属性计算出重要度 $\mathrm{SGF}(C-\{c_i\},D)$ 为 0 时，进行对属性 $c_i$ 的约简变换。

该变换规则元知识高度概括了属性约简的原理和本质。

**2. 数据挖掘的变换规则元知识**

数据挖掘是从大量数据中获取知识，这些知识实质上是这些数据的高度浓缩，仍保留了数据的本质。这里讨论基于粗糙集理论的数据挖掘方法的元知识。

该方法是通过条件属性集 $E_i$ 与决策属性集 $Y_j$ 之间的上下近似关系来获取知识。为此要建立一个求解两集合 $E_i$ 和 $Y_j$ 之间上下近似关系的算子 $A_{updow}$。

(1) $A_{updow}(E_i,Y_j)$算子的计算过程

① 求条件属性集 $C$ 中的等价类 $E_i$；

② 求结论属性集 $D$ 中的等价类 $Y_j$；

③ 求 $E_i$ 和 $Y_j$ 之间的交，分别有三种情况：

① $E_i \cap Y_j = E_i$； ② $E_i \cap Y_j \neq E_i(\neq \varnothing)$； ③ $E_i \cap Y_j = \varnothing$

该算子对前两种情况能生成两类规则知识，它实际上是从数据库 $D_{set}$ 中获取规则知识的数据挖掘变换，表示为

$$T_{DM}(D_{set}) = (E_i \to Y_j)$$

(2) 基于粗糙集的数据挖掘方法的变换规则元知识

数据挖掘的变换 $T_{DM}$ 是算子 $A_{updow}(E_i, Y_j)$ 的计算结果引起的，从中可得到两条变换规则元知识：

$$(A_{updow}(E_i, Y_j) = E_i) \to (T_{DM}(D_{set}) = (E_i \to Y_j)) \tag{12.33}$$

$$(A_{updow}(E_i, Y_j) \neq E_i) \wedge (A_{updow}(E_i, Y_j) \neq \varnothing) \to T_{DM}(D_{set}) = ((E_i \to Y_j), Cf) \tag{12.34}$$

其中可信度 $Cf$ 为：$Cf = |E_i \cap Y_i| / |E_i|$

这两条变换规则元知识高度概括了粗糙集获取知识的原理和本质。

#### 12.1.5.3 专家系统的变换规则元知识

专家系统中的元知识主要用来对专家系统运行的控制，用变换规则知识来表示控制专家系统运行的元知识是很合适的。专家系统一般采用逆向推理，它运行控制的元知识主要包括：指定目标开始推理；检查当前变量是否处于推理树的叶结点，若是则进行提问；提问回答符合要求时，推理进行回溯；提问回答不符合要求时，继续提问；目标求出值后，停止推理或转向另一推理树的目标等。下面对其中部分元知识利用变换规则表示形式，更能体现变化的特点：

(1) 叶结点提问处理：当推理过程中发现当前结点 $x$ 是叶结点 $x_0$ 时，将叶结点变换成给定叶结点的提问句，元知识表示为

$$\text{Compare}(x, x_0) = \text{yes} \to T_{ques}(x_0) = (\text{question}(x_0) = \text{“提问句”})$$

(2) 叶结点用户回答正确处理：当用户回答的值 $v(\text{user})$ 属于叶结点取值 $v(x_0)$ 的范围时，推理进行回溯，即将上层结点 $x'$ 置换叶结点 $x_0$，元知识表示为

$$\text{Compare}(v(\text{user}), v(x_0)) = \text{yes} \to T_{repl}(x_0) = x'$$

(3) 叶结点用户回答不正确处理：当用户回答的值 $v(\text{user})$ 不属于叶结点取值 $v(x_0)$ 的范围时，则继续提问，元知识表示为

$$\text{Compare}(v(\text{user}), v(x_0)) = \text{no} \to T_{ques}(x_0) = (\text{question}(x_0) = \text{“提问句”})$$

(4) 单推理树推理控制：当目标结点 $G$ 通过推理求出值 $v_G$ 时，停止推理。元知识表示为：

$$\text{Check}(v(G) = v_G) \to T_{stop}(x = G) = R_{stop}$$

(5) 多推理树推理控制：当一个推理树的目标结点 $G$ 通过推理求出给定值 $v_G^*$ 时，控制推理机从该推理树转向另一推理树 $i$ 的目标结点 $G_i$，元知识表示为

$$\text{Check}(v(G) = v_G^*) \to T_{repl}(G) = G_i$$

用变换规则知识，即变换产生式式来表示专家系统中的元知识，比原来采用的元知识表示更能体现变化的特点，也便利了专家系统程序，容易控制专家系统有效运行。

#### 12.1.5.4 结束语

通过以上研究可知，神经网络、知识发现都是一个过程，需要经过若个步骤来完成，用变换规则作为元知识来描述，既适应了求解过程的变化需求，又起到了把定量问题进行定性化

描述，即浓缩了具体的定量计算过程的效果。在专家系统中的元知识用变换规则表示，更突出了运行专家系统的控制效果。

变换规则作为元知识的一种新的表示形式，是对元知识的扩充，既能有效地把握问题的本质，又能有效地起到指导和控制系统运行的效果。变换规则作为元知识的表示形式，能够适应变化的环境，具有广泛的应用前景。

## 12.2 软件进化规律的知识挖掘

计算机虽然是非生物，但在人的帮助下，它解决问题的能力充分体现了由简单到复杂、由低级到高级这种进化过程。这种进化过程的结果，使计算机逐渐在向人靠拢，逐步在代替人的智力工作。找出计算机进化的规律，一来是为了提升人们利用计算机解决问题的能力，二来是为了促进计算机的进一步进化。

计算机软件的进化主要经历了：①数值计算的进化；②计算机语言的进化；③从“数值计算”到“数据处理”再到“知识推理”的进化等。

### 12.2.1 数值计算的进化

数值计算的进化体现在从“算术运算”到“微积分运算”再到“解方程”的发展过程。

#### 1. 数值计算能力的进化

数值计算能力的进化概括为(说明：“→”表示进化，“←” 表示回归)：

$$+\rightarrow\pm\times\div\rightarrow\text{初等函数}\rightarrow\text{微积分运算}\rightarrow\text{解方程}$$

即“+” 运算是数值运算的根本。

(1) $\pm\times\div\leftarrow+$

① 加(+)是最基本运算。

② 减(−)是利用减数的补数(求反加 1)，变减为加。

③ 乘(×)是把乘变成累加，如 $5\times3=5+5+5$，即 5 加 3 次。

④ 除(÷)是把除变成累减的次数，如 $6\div3$ 为 $6-3-3=0$，减了 2 次，即商为 2。

(2) 初等函数$\leftarrow\pm\times\div$

初等函数的计算不是利用定义，而是利用台劳级数公式来计算的，即变成(回归到)加减乘除运算。如：

① 三角公式

$$\sin x=\frac{x}{1!}-\frac{x^3}{3!}+\frac{x^5}{5!}-\frac{x^7}{7!}+\cdots$$

② 指数公式

$$e^x=1+\frac{x}{1!}+\frac{x^2}{2!}+\frac{x^3}{3!}+\cdots$$

注意：取级数的项数在于满足计算的精度。

(3) 微积分运算 $\leftarrow\pm\times\div$

① 微分运算(差分化)

$$f'(x)=\lim_{\Delta x\to 0}\frac{f(x)-f(x_0)}{x-x_0}\quad 变成\quad f'(x)\approx\frac{f(x)-f(x_0)}{x-x_0}$$

即导数的极限运算变成近似的差分求商，也就是回到了加减乘除运算。

② 积分运算(求和)

$$\int_a^b f(x)\mathrm{d}x=\lim_{\Delta x\to 0}\sum_{k=1}^{n}f(x_k)\Delta x\quad 变成\quad \int_a^b f(x)\mathrm{d}x\approx\sum_{k=1}^{n}f(x_k)\Delta x$$

即积分的极限运算变成近似的求和，也回到了加减乘除运算。取 $\Delta x$ 尽量小，就能满足误差精度。

③ 二阶导数的差分方程

$$\frac{\mathrm{d}^2 f(x)}{\mathrm{d}x^2}=\frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{\mathrm{d}f(x)}{\mathrm{d}x}\right)\approx\frac{f(x_2)-2f(x_1)+f(x_0)}{\Delta x^2}$$

一阶和二阶导数的结点关系如图 12.3 所示。

高阶导数处理方法类似。

④ 偏微分方程的差分方程

$$\frac{\partial u}{\partial y}+\frac{\partial^2 u}{\partial x^2}\approx\frac{u_n^{n+1}-u_{n+1}^n}{\Delta y}+\frac{u_{j+1}^n-2u_j^n+u_{j-1}^n}{\Delta x^2}$$

说明：$n$ 表示 $y$ 方向的增长，$j$ 表示 $x$ 方向的增长。偏导数结点关系如图 12.4 所示。

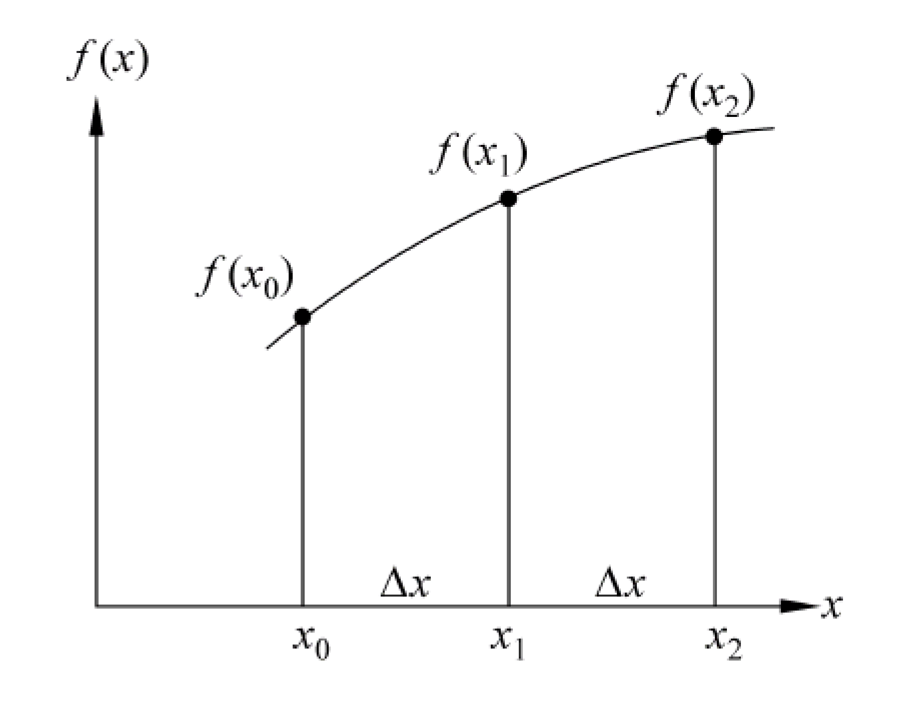



图 12.3　一阶和二阶导数的结点关系

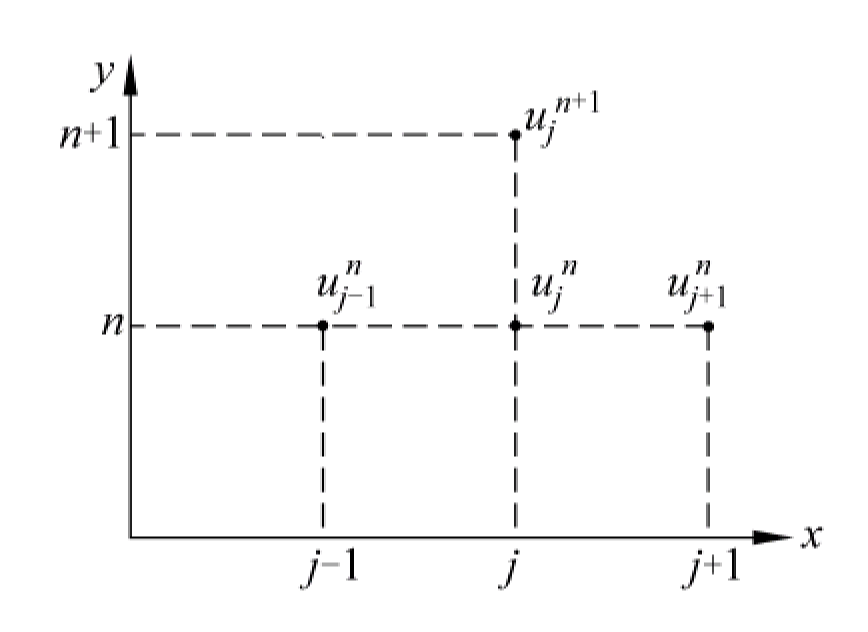



图 12.4　偏导数结点关系

(4) 解方程

方程的求解有两种方法：直接求解法和迭代求解法。

① 方程的直接求解

- 线代数方程组的直接求解

线代数方程组的结构形式一般表示(人理解的方式)为

$$a_{11}x_1+a_{12}x_2+\cdots+a_{1n}x_n=b_1$$
$$a_{21}x_1+a_{22}x_2+\cdots+a_{2n}x_n=b_2$$
$$\cdots\cdots$$
$$a_{n1}x_1+a_{n2}x_2+\cdots+a_{nn}x_n=b_n$$

在计算机中，方程组用矩阵(数组)形式表示为

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}, \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

说明：计算机中并不存在方程的结构形式，分别用三个数组表示，它们可以存放在计算机中不同的地方。这种表示把运算符（×、+、=）都隐藏起来，这有利于同类数据集中存储，运算符将体现在指令操作中，即计算机程序把数据和运算符分开了，这是计算机程序的重要特点。

解方程时只对三个数组进行处理，最后得出 $x_i$ 值。

线代数方程组的高斯主元素消去法（加减乘除）：系数矩阵消元成单位矩阵

$$\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}, \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, \begin{pmatrix} b'_1 \\ b'_2 \\ \vdots \\ b'_n \end{pmatrix}$$

- 偏微分方程边值问题的求解

偏微分方程边值问题的求解一般是在一个区域内进行，区域中的点是未知数，区域边界点是已知数。例如汽轮机转子进行热传导偏微分方程的计算，其网络划分如图 12.5 所示。

偏微分方程差分化后，经过整理就变成了以区域中的点为未知数，区域边界点是已知数的线代数方程组。

偏微分方程的求解就变成了线代数方程组的求解，即回到了加减乘除的运算。

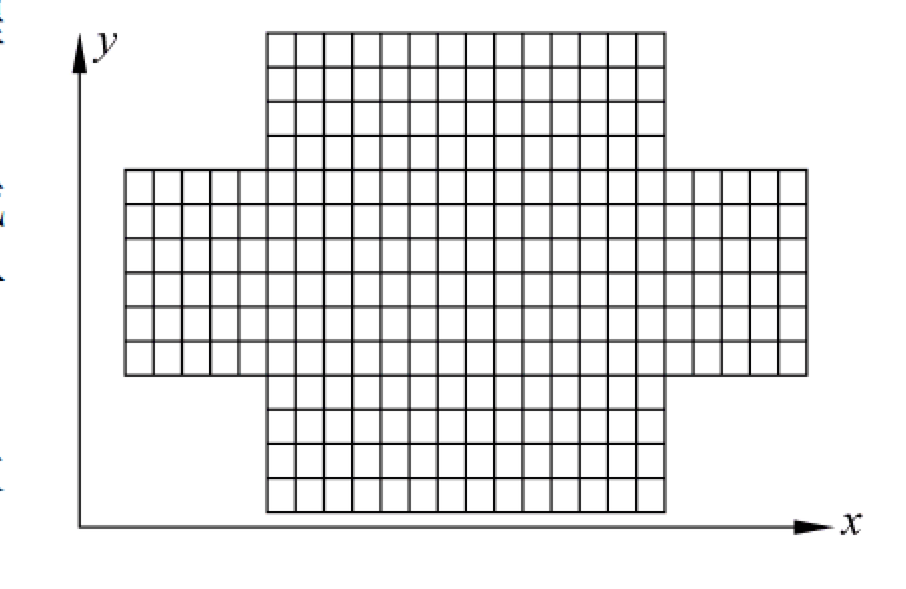


图 12.5　汽轮机转子的网络划分

- 微分方程数值计算的价值

传统的数学分析解方程的方法是通过推演得到解析解，即用表达式形式表示的解。求方程的解析解只能解决少数的较简单的和典型的微分方程的求解。

微分方程的数值计算方法，无论是常系数还是变系数，是线性还是非线性，都能得到解决。解决的手段是对微分方程差分化，得到差分方程，让计算机来解差分方程（加减乘除）得到数值解。

② 方程的迭代求解

- 迭代法的思想

将方程 $f(x)=0$ 变成 $x=\varphi(x)$

建立迭代法方程：

$$x_{n+1} = \varphi(x_n) \quad n = 1,2,\cdots,\infty$$

初值 $x_1$ 任意选定，经过无限次迭代后，使

$$|x_{n+1} - x_n| \leqslant \varepsilon$$

这时，$x_n$ 就是原方程的解：

$$f(x_n) = 0$$

典型的是牛顿迭代公式是：

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad n = 1,2,\cdots,\infty$$

牛顿迭代公式中的导数是切线，经过多次迭代，很快就能求得方程的解 $x_n$。

- 迭代方法适合于计算机求解

用迭代方法求解方程使解方程更简单和容易，省去了烦琐的步骤，思路简单。迭代方法很适合让计算机来完成。因为迭代次数一多，人来做就无法实现，而计算机来做就不成问题。计算机运算速度很快，适合重复性的计算。

计算机为迭代方法求解方程开辟了新路。

- 迭代公式的讨论

迭代公式的计算结果有两种可能：收敛（求得结果）和发散。

当发散时需要构造反函数，才能使迭代收敛，即

$$x = \varphi^{-1}(x)$$

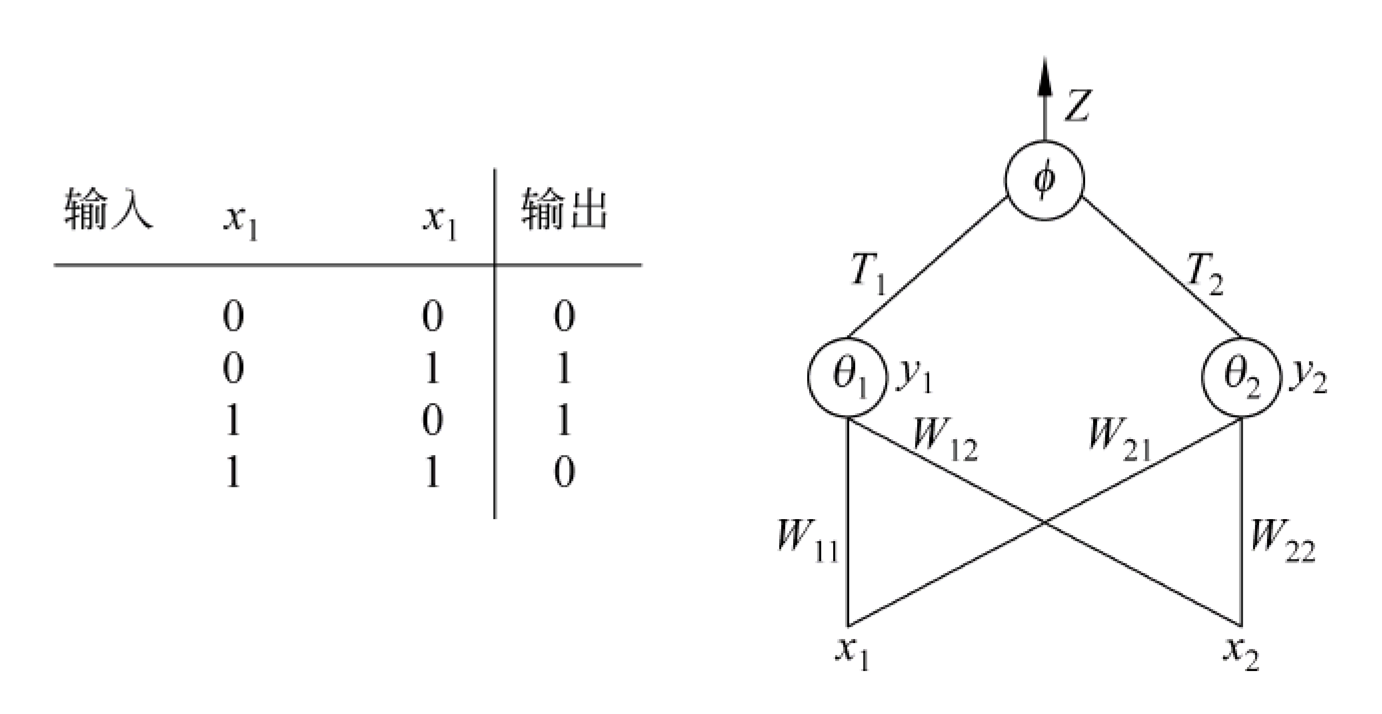
按以上思路，我们可以先将原方程构造迭代公式，不行时就构造反函数。

- 迭代法的典型实例

BP 神经网络中权值和阈值的求解就是采用迭代法，具体公式为

$$W_{ij}(k+1) = W_{ij}(k) + \eta'\delta'_i x_j$$
$$\theta_i(k+1) = \theta_i(k) + \eta'\delta'_i$$

例如：异或问题的 B-P 神经网络如图 12.6 所示。



| 输入 $x_1$ | $x_1$ | 输出 |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |



图 12.6　异或问题的 B-P 神经网络

计算机运行结果；迭代次数：16745 次；总误差：0.05

隐层网络权值和阈值：

$$w_{11} = 5.24,\quad w_{12} = 5.23,\quad w_{21} = 6.68,\quad w_{22} = 6.64,\quad \theta_1 = 8.01,\quad \theta_2 = 2.98$$

输出层网络权值和阈值：

$$T_1 = -10,\quad T_2 = 10,\quad \phi = 4.79$$

(5) 数值计算的误差问题

数值计算的误差积累会引起结果的错误。例如，应该是正数的，但计算出来的是负数。为了使误差积累不产生错误的结果，需要：

① 原始数据的有效位数要比结果数据有效位数多出 1～2 位。

② 使用不很合理的公式时，要检查可能出现错误的地方，增加判别公式，防制错误发生。

**2. 二进制计算到二值数据表示**

(1) 二进制数从计算到表示

二进制数开始在计算机中是用于计算（代替十进制计算），后来发展成为表示形式（如汉字、图像、声音等）。概括为：

十进制计算→二进制计算→二值数据表示→汉字编码＋字形的二值数据表示（点阵）→图像点阵表示

从二进制数值计算到用二值数据表示汉字或多媒体，是一次重大的观念转变。它使汉字或多媒体能够存入计算机中，也就可以在计算机中进行处理。这使计算机扩大了它的处理范围，使计算机进入了多媒体时代，也标志了计算机向前迈进了一大步。

(2) 十进制到二进制的转换

计算机只能采用二进制。在使用计算机进行数值计算时，虽然输入的数是十进制数，但在计算机内有一个子程序（类似于初等函数子程序）会把数据转换成二进制。

(3) 汉字表示

① 英文字母、数字、标点符号等用 ASCII 码值表示

如 A 的码值 65，数字 0 的码值 48。

② 汉字编码

一个汉字用 4 位十进制数字编码，前两位是区号，后两位是位号。一个汉字在计算机中的内码占两个字节，第一个字节用于区号，第二个字节用于位号。

汉字的形状是方块体的多笔画的字，采用了二值数据的点阵形式来表示。这就使计算机能存储汉字，并能处理汉字。这克服了计算机只能处理拼音文字的狭隘的范围，使汉字也能用计算机这个现代工具来处理。这既促进了汉字文化的发展，又使计算机的处理能力上升了一步。

(4) 图像的表示

图像看成点（像素）的集合，每个像素的颜色用三个字节（24 位）表示。任何颜色由红、绿、蓝三色混合而成，三色各占一个字节，一个字节中各位的 0 或 1（二值数据）的不同来表示，构成了不同的颜色浓度。

一幅图像在计算机中表示为一个长度惊人的 0、1（二值数据）串。

图像用点阵数据表示，使计算机就能存储图像，并能处理图像。从而使计算机进入了多媒体时代。

(5) 视频的表示

视频是连续播放一系列图像。每幅图像称为帧。每秒播出帧的数目在 24～30 幅图像时，就是像电影一样的视频。

由于视频数据量太大，一般采取 MPEG 压缩技术，相邻帧只记录前面帧的变化部分。不记录前面帧的重复部分，这样就可以节省大量的存储空间。

### 12.2.2 计算机程序的进化

计算机程序的进化可以概括为：

二进制程序→汇编程序→高级语言程序→程序生成

#### 1. 二进制程序

(1) 机器语言(二进制) 程序

二进制程序是最原始的计算机运行程序，由一串机器指令组成。

机器指令含：操作码和地址码。

| | | | | |
|---|---|---|---|---|
| 例如：操作码： | 02 | 加法 | 地址码：1001 | x |
| | 05 | 取数 | 1002 | y |
| | 06 | 送数 | 1003 | 空 |

完成 x+y 的计算程序(八进制)为：

```
05  1001  取 x
02  1002  加 y
06  1003  送结果
```

机器语言程序有两个重要特点：

① 在地址码中只放数据，不放运算符。运算符都在操作码中，即运算和数据是分开的。

② 操作码中的指令是对变量的地址进行操作，而不是直接对变量的操作。这是一种间接操作，这适合机器的运算。

因为在对变量进行运算之前，先要对变量进行存储，即把变量放入某个地址单元中，要进行运算就必须从地址单元中取出变量，再进行计算，指令对地址进行操作，就是完成这些动作，这就形成了间接操作。

间接操作的好处在于：

① 对于不同数据的相同操作，只需把不同数据放入相同地址单元中，程序不用变化。间接操作为程序的通用性带来了好处。它有别于人对变量的直接操作。人操作时，不需要把数据放入某个地址单元这个动作。

② 编程序时，不要求先把数据都准备好后再编程序，只需把数据的存放地址都分配好后就可以编程序。

我国 20 世纪 60 年代研制的第一台计算机(电子管)103 型(仿苏联的 M3)，以及后来的 104、109 型等多台计算机，提供的都是机器语言(二进制)。

(2) 汇编程序

汇编程序是将二进制(或八进制、十六进制)程序中的数字用字母符号(助记符)代替。使用汇编程序简化了繁琐的数字。

上例程序的汇编程序为：

```
LDA   x   取 x
ADD   y   加 y
STA   r   送结果
```

汇编程序便利人书写，虽然程序中书写是变量 x、y，但是，汇编程序运行时还是要返回到二进制程序。程序中的变量仍然要用它的地址单元来表示。这时，变量的地址单元是由机器的解释程序来分配的。它不同于人编制的二进制程序，变量的地址单元是程序员分配的。

汇编程序通过解释程序返回到二进制程序。解释程序很简单，只需要二张对照表即可，一个是指令操作码的二进制对照表，另一个是数据地址的二进制对照表。

(3) 高级语言程序

高级语言程序是用接近自然语言和数学语言编写的程序。接近人们的习惯，便利非专业人员编写。高级语言程序种类很多，完成数值计算的高级语言有 C、Pascal、ADA 等；完成数据库操作的高级语言有 FoxPro、Oracle、Sybase 等；完成知识推理的高级语言有：PROLOC、LISP 等。高级语言程序需要先对所有的数据元素（变量、数组等）都要指定清楚，便利编译程序分配地址单元，即高级语言程序仍然是对数据的间接操作。

高级语言把程序的运算能力提高了一大步，即高级语言的结构化程序设计中，把程序结构归纳为三种基本结构的组合，这三种基本结构是顺序、选择、循环。任何复杂的程序都是这三个基本结构的嵌套组合。这种程序结构保证了程序的正确性。这在“程序设计方法学”中给出了正确性的证明。它克服了 20 世纪 60 年代的软件危机。

在机器语言的指令集中，有比较和转移（Go To）指令，也能完成选择和循环的运算，但当时程序员有一种追求编写精巧程序的愿望，于是大量使用 Go To 语句。对于一个小程序是一个精巧程序时，它是一个艺术品；对于一个大程序，在大量使用 Go To 语句以后，发生错误的概率将大大增加，这就成了灾难，形成了软件危机。当时，不少人提出取消 Go To 语句。最后，由于提出了结构化程序设计思想才解决了这场软件危机，使大型程序的正确性得到了极大的提高。

高级语言的效果体现在：

① 高级语言便利了程序的编写；

② 高级语言的功能更强了（很多标准的程序段通过连接程序直接嵌入到用户程序中），极大地提高了解决问题的能力和扩充了计算机的应用范围；

③ 高级语言的应用促进了新语言的出现，面向对象语言、数据库语言、网络编程语言以及第四代语言（程序生成）等陆续出现。

**2. 高级语言程序的编译**

(1) 编译程序的思想

高级语言程序同样要返回到二进制程序，这就是编译程序。

编译程序包括词法分析、语法分析、代码生成。它的技术原理相同于人工智能中的专家系统。即利用文法（知识）对程序中的语句进行归约（反向推理）或推导（正向推理），既要检查语句是否合符文法，又要将语句编译成中间语言或机器语言。

计算机程序的本质还是二进制程序。

转换过程如下：

源程序 →（编译程序）→ 二进制程序

（2）表达式的编译

表达式的编译是编译程序中最复杂的部分。人进行表达式计算时要按照规定进行：先乘除，后加减，括号优先。计算机对表达式的计算，不能按此规定进行，因为这不便于编制程序来适应这种规定。

表达式的编译采用了波兰逻辑学家 J. Lukasiewicz 1951 年提出的逻辑运算无括号的记法：①前缀表达式——波兰式；②后缀表达式——逆波兰式。

也就是将人习惯的中缀表达式变成后缀表达的逆波兰式，逆波兰式把表达式中的括号去掉了，把加减乘除的优先级别变成了前后的顺序关系，这就适合计算机的顺序处理。例如：

```
u* v+p/q→uv* pq/+
a* (b+c)→abc+ *
```

在“编译程序”书中，将中缀表达式变成后缀表达的逆波兰式，占了很大的篇幅。一般利用一个符号栈或者采用递归子程序的方法来完成这种转变。

### 12.2.3 数据存储的进化

数据存储的进化可以概括为：

变量→数组→线性表→堆栈和队列→数据库→数据仓库

#### 1. 数据存储的进化过程

（1）变量→线性表

① 变量：计算公式中的基本元素，分配一个存储地址。

② 数组：相同类型的一维、二维数据集合，存储地址是连续的。

③ 线性表：不同类型数据的集中存储。如学生表中含姓名、性别、年龄等不同类型的数据集合。

（2）堆栈和队列

它是指用于特殊运算而暂时存放的数组或线性表。

① 堆栈。对进栈的数据采用后进先出的处理方式，如对急诊病人的处理：后来的先看病。

② 队列。对进队的数据采用先进先出的处理方式，如对一般病人的处理：按排队先后顺序看病。

（3）数据库

通过数据库管理系统管理的数据文件。

数据库管理系统（数据库语言）的主要功能为：

① 建立数据库。描述数据库的结构并输入数据。

② 管理数据库。a 控制数据库系统的运行；b 进行数据的检索、插入、删除和修改的操作。

③ 维护数据库。a 修改、更新数据库；b 恢复故障的数据库。

④ 数据通信。完成数据的传输。

⑤ 数据安全。设置一些限制，保证数据的安全。

数据库存储结构不同于数组，数据库的存储结构由两大部分组成：文件头部分和记录正文部分。

文件头部分包括数据库记录信息和各字段的说明。数据库记录信息是由年月日、记录数、文件头长度、记录长度等信息组成。0DH 和 00H 两字节为文件头的尾，如图 12.7 所示。

| |
|---|
| 数据库记录信息(32 字节) |
| 字段说明 (32 字节) |
| … … … |
| 0DH 00H |
| 记录正文部分 |
| … … … |
| … … … |
| … … … |

图 12.7　数据库的存储结构

记录正文部分的存储结构如图 12.8 所示。

| | | | | |
|---|---|---|---|---|
| 记录 1 | 删除标志(1B) | 第 1 字段内容 | …… | 第 $N$ 字段内容 |
| 记录 2 | …… | | | …… |
| 记录 3 | …… | | | …… |

图 12.8　记录正文部分的存储结构

每个记录增加一个删除标志在于删除该记录时，只做删除标志，并没有真正抹去该记录。这样使记录的索引不发生变化，不影响整个数据库的其他操作。增加删除标志，虽然多了冗余，但便利了数据库的操作。

数据库的数据存储量大小不一，一般在 100MB 左右。

(4) 数据仓库

数据仓库是大量数据库(二维)集成为多维数据的集合，如图 12.9 所示。

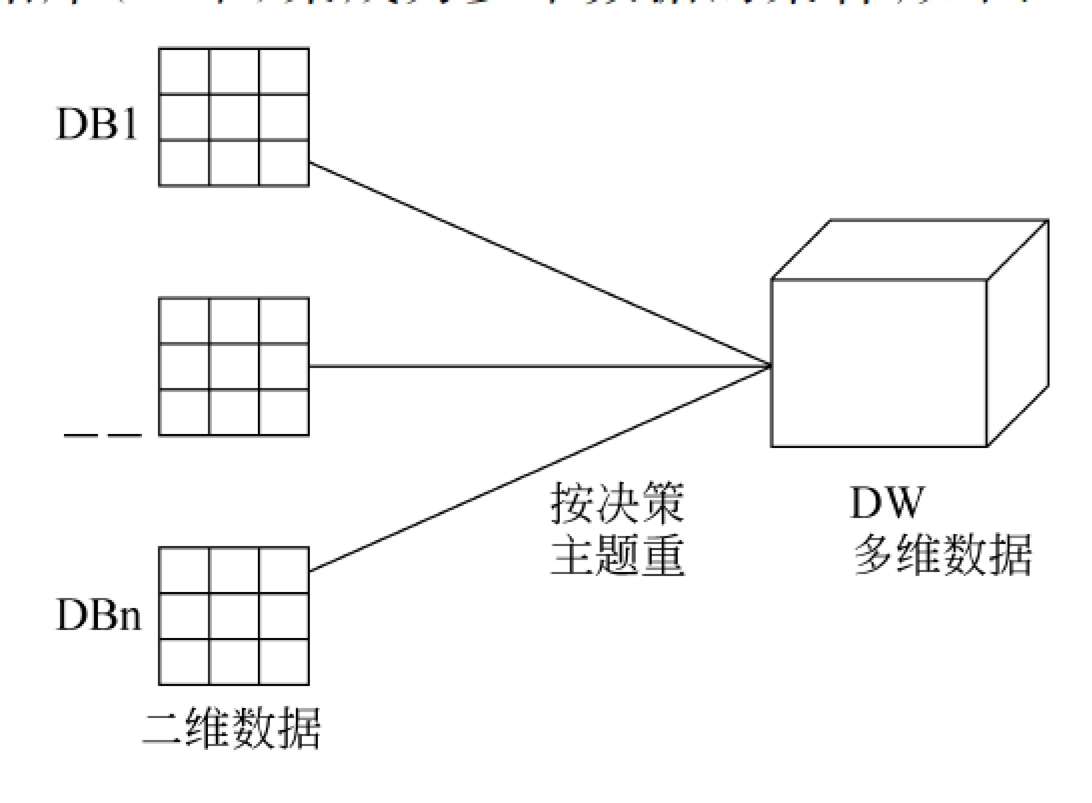



图 12.9　由数据库形成数据仓库的示意图

数据仓库中的数据分为多个层次，包括当前基本数据层、历史数据层、轻度综合数据层、高度综合数据层、元数据。数据仓库结构如图 12.10 所示。

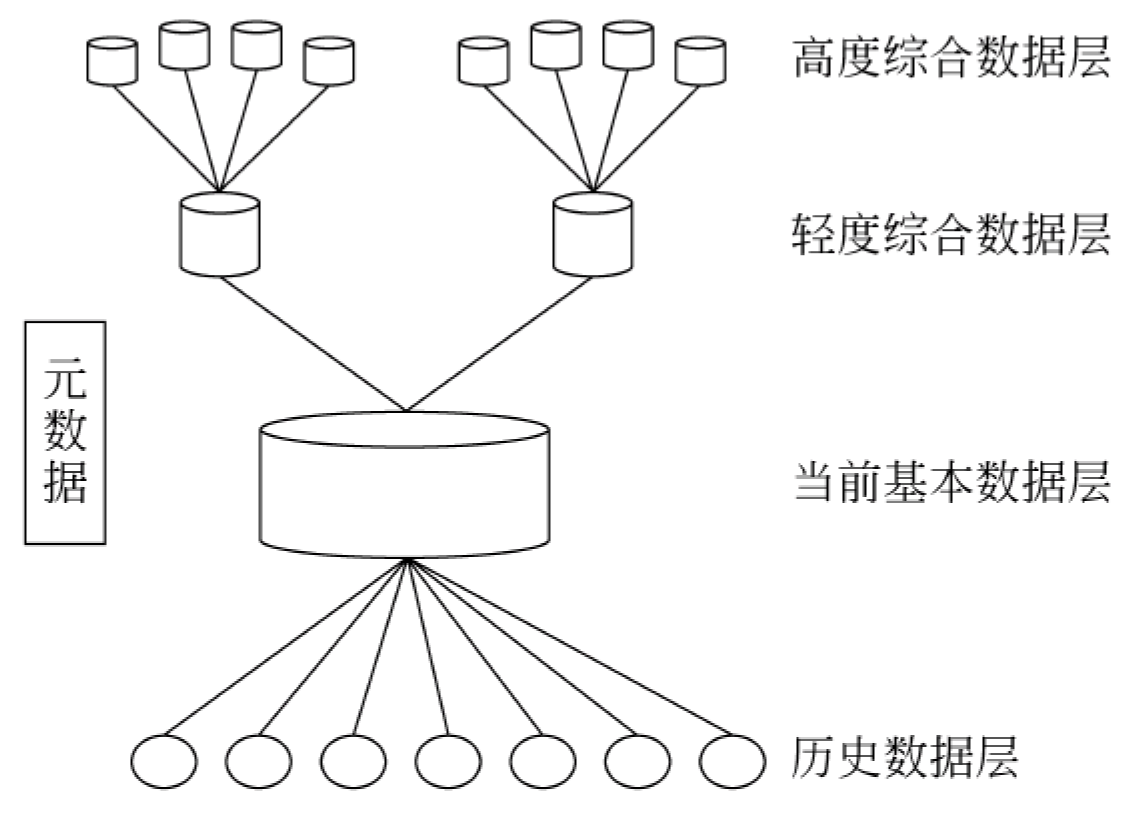



图 12.10　数据仓库中的数据层次结构

由于数据仓库的数据是多维数据，数据仓库的存储结构采用了“星型模型”。星型模型是由“事实表”(大表)以及多个“维表”(小表)所组成的。“事实表”中存放大量关于企业的事实数据。“维表”(相当于多维坐标系中的坐标维的数据)中存放坐标维的描述性数据，维表是围绕事实表建立的较小的表。每个表均采用关系数据库的存储结构形式。

一个星型模型数据的实例如图 12.11 所示。

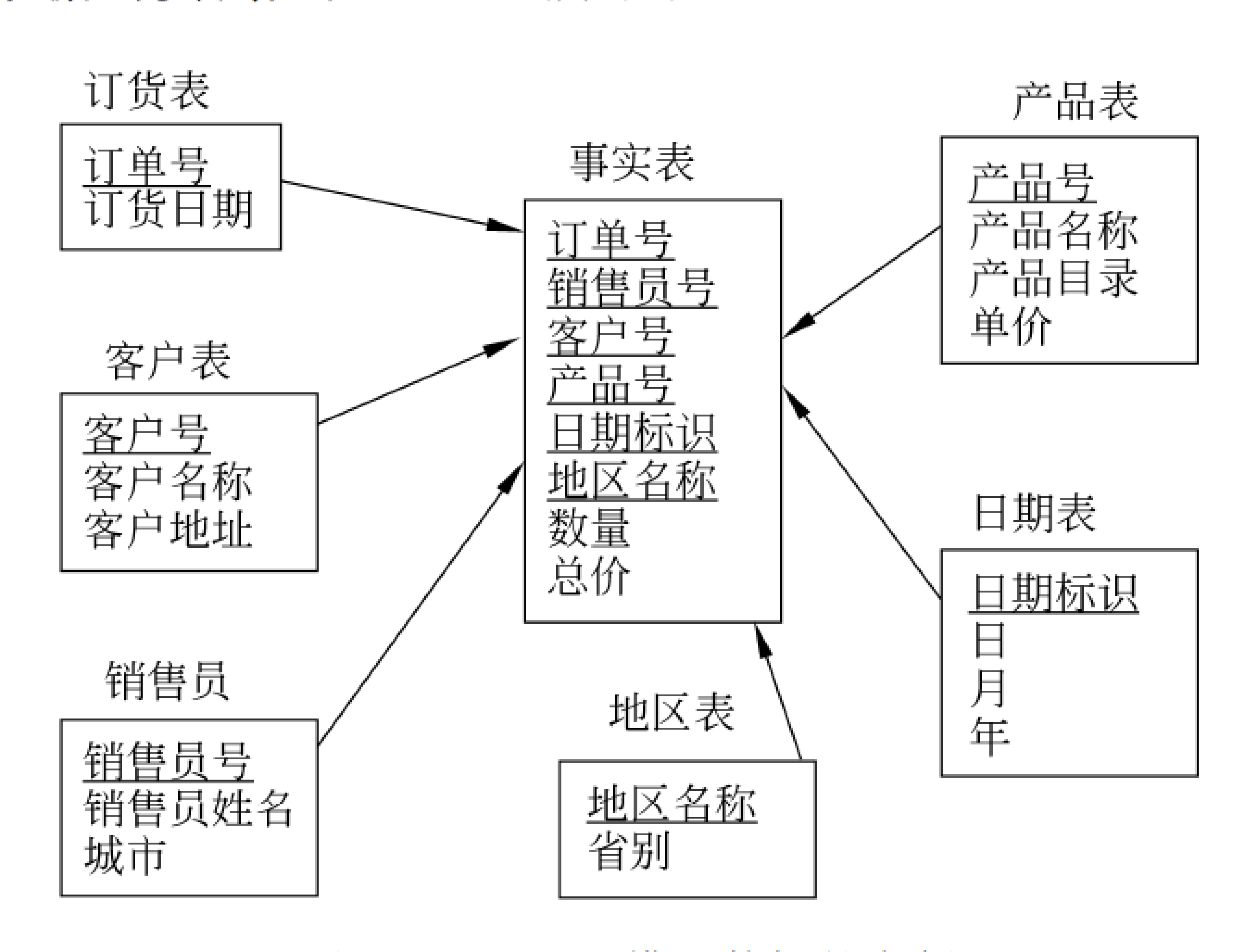



图 12.11　星型模型数据的实例

数据仓库的数据存储量一般在 10GB 左右，它相当于数据库的数据存储量的 100 倍。大型数据仓库的数据存储量达到了 TB(1000GB)级。这种数量级的数据存储，只有在计算机发展到今天的水平，存储量的飞速剧增才能实现。

**2. 用于管理的数据库和用于决策的数据仓库**

(1) 用于管理的数据库

数据库一般只存储当前的现状数据，用于管理业务(商业计算)。数据库的特点是：

① 不同的业务(人事、财务、设备等)需要建立不同的数据库；

② 随时间、业务的变化随时修改数据；

③ 数据库是共享的数据。

由于数据库的出现使计算机走向了社会。现在社会中的各行各业已经离不开数据库了，数据库已成为各行各业现代化管理的基础设施。

(2) 用于决策的数据仓库

决策需要大量的数据。有了数据仓库以后，计算机利用数据辅助决策成为现实，因为数据仓库中存储了当前数据、历史数据和汇总数据。辅助决策的方式主要有：

① 历史数据用于预测；

② 从汇总数据的比较(不同角度)中发现问题；

③ 从详细数据中找出原因。

**3. 数据存储进化的小结**

计算机的数据存储量愈来愈大，数据种类也愈来愈多，这样使计算机处理问题的能力也愈来愈强。

数组一般用于数值计算，数据库用于管理业务，数据仓库用于决策支持。

数据是计算机解决实际问题的基础。数据存储是计算机重要组成部分，数据存储的进化是计算机进化的一个大的方面。

## 12.2.4 知识处理的进化

知识处理的进化中，一个典型过程可以概括为：

知识表示与知识推理→专家系统→知识发现与数据挖掘

**1. 知识表示与知识推理**

(1) 知识表示

知识在计算机中的存储和使用的形式，典型的知识表示有：

产生式规则($A \rightarrow B$)、谓词 $P(x,y)$ 等

(2) 知识推理

从已知条件利用知识推出结果：

规则的推理：假言推理：$p \rightarrow q, p \vdash q$

谓词的推理：归结原理 (反证法)

(3) 谓词推理例

谓词逻辑是用谓词公式表示文本内容。

例：每个储蓄的人都获得利息。表示成谓词公式为：

$$\forall x[(\exists y)(S(x,y)) \wedge M(y)] \rightarrow [(\exists y)(I(y) \wedge E(x,y))]$$

其中：$x$ 表示人，$y$ 表示钱，$S(\ )$表示储蓄，$M(\ )$表示有钱，$I(\ )$表示利息，$E(\ )$表示获得。

谓词的推理分两部分：①把谓词公式化简成只含 $\vee$ 的子句(包括～)；②归结。

谓词公式中包含所有逻辑运算符，即 $\wedge$　$\vee$　～　$\rightarrow$　$\leftrightarrow$ 和 $\exists$　$\forall$ 。化简过程主要有：

①消去：→、∃、∀；②把谓词公式化为合取范式；如$(A \vee B) \wedge (C \vee D)$；③分解合取范式为只含∨的子句。该子句变为$A \vee B$、$C \vee D$。

上面谓词公式的子句为：

① $\sim S(x,y) \vee \sim M(y) \vee I(f(x))$

② $\sim S(x,y) \vee \sim M(y) \vee E(x,f(x))$

对于谓词逻辑的推理的归结原理(反证法)是利用前提谓词公式证明结论谓词公式：

① 把前提谓词公式化简成子句。

② 把结论谓词公式取非后化简成子句。

③ 归结时，消去二个子句中正、负谓词后合并为一个子句。

④ 归结的最后为空子句(产生矛盾)，就证明了结论谓词公式的正确性。

(4) 知识推理不同于数值计算

知识推理使计算机进入符号处理的新领域。这种符号处理是建立在逻辑运算的基础上，逻辑运算符号有多个。在谓词逻辑中，对谓词公式要化简成只含∨的子句(包括～)，这样就大大简化了归结运算。在归结中需要找正、负子句，这少不了一个"对比"操作，在计算机的指令中，有"比较"操作。

可以看出，"比较"操作是逻辑运算的基础。

### 2. 专家系统

专家系统中对规则知识的逆向推理，并没有将所有的规则都连接成一棵知识推理树，进行深度优先搜索。而是利用规则栈，反复地搜索知识库中的知识，通过知识的进栈和出栈，达到推理树的深度优先搜索。为什么要这样做?

理由有两个：①将规则知识连接成知识推理树并不好做，因为树的分支个数是不固定形式的，用指针链表难于设计；②在规则栈中从栈顶规则知识找到和它连接的知识，需要在知识库中从头到尾搜索一遍知识库，才能找到所要的知识。同样，继续找下一个连接的知识，又得在知识库中从头到尾搜索一遍知识库，才能找到所要的知识。这种反复搜索知识库中知识的操作，对计算机程序而言是很容易的，可利用循环来完成。

虽然，知识推理采用规则栈的方式是合适的，这是用耗费计算机的计算时间(反复搜索知识库)来完成知识的推理。

知识库中搜索找到所要的知识，也是一个"比较"操作。可见，"比较"操作对于规则知识的推理和谓词推理的归结都是基础。可以归纳出，"比较"操作是符号处理的基础。

### 3. 知识发现与数据挖掘

知识发现与数据挖掘已经在第6章中做了详细说明。这里只讨论粗糙集方法的属性约简和分类知识的获取的逻辑计算基础。

粗糙集以等价关系(不可分辨关系)为基础，用于分类问题。等价关系定义为，不同元组(对象)$x$和$y$对属性$a$的等价关系是它们的属性值相同。等价类是所有具有等价关系的对象的集合。粗糙集定义了上、下近似两个集合来逼近任意一个集合$X$。上近似定义为：等价类中元素$x$都属于$X$。下近似定义为：等价类中元素$x$可能属于$X$，也可能不属于$X$。

（1）粗糙集的属性约简方法

粗糙集的属性约简原理是：在条件属性集 $C$ 中去掉一个条件属性 $c$ 后，相对于决策属性 $D$ 的正域与去掉属性 $c$ 前的正域相同，该属性 $c$ 可约简。

计算正域时需要进行等价类计算，等价类的计算就是要对属性值进行“比较”操作，检查是否相同。

可见，“比较”操作是属性约简方法的基础。

（2）分类知识的获取

粗糙集的分类知识获取原理是依据集合的蕴含关系，当条件属性集中的等价类蕴含于决策属性的等价类，则存在它们之间的分类规则知识。这种蕴含关系若是上近似，则分类规则知识的可信度为 1。这种蕴含关系若是下近似，则分类规则知识的可信度小于 1。

蕴含关系的计算涉及集合域的比较，即条件属性集中的等价类与决策属性的等价类的比较。

可见，“比较”操作也是分类知识获取方法的基础。

知识发现与数据挖掘的其他方法中的逻辑计算都是以“比较”操作为基础的。

### 12.2.5 进化规律的知识挖掘

#### 1. 计算机的原始本能

通过以上分析，首先要总结一下计算机的原始本能。它主要包括如下三点。

（1）数值计算的加法

任何复杂的运算只要能化简成算术运算（加、减、乘、除），它就能在计算机中进行运算，如微积分计算、解方程等都要化简成算术运算。算术运算又可归结为加法运算。

（2）二值数据表示

二进制数据开始时用于计算，后来发展为用二值数据来表示。任何媒体只要能用二值数据表示，它就能在计算机中存储和处理。这是计算机存储的基础。

（3）逻辑运算的比较

数值计算、数据管理、知识处理等中间的逻辑运算的本质是“比较”操作，数值的比较是大小的比较，符号的比较在于是否相同。计算机程序的顺序、选择、循环结构的运行基础也是逻辑运算的“比较”。

#### 2. 计算机的优势和不足

（1）计算机的优势

① 计算机的存储量很大

计算机的飞速发展使计算机的存储量愈来愈大。这样，使汉字、多媒体能以大量的二值点阵数据存入计算机中。使计算机既能求解未知数上万的方程组，也能处理变化多端的多媒体。

② 计算机的计算速度很快

计算机的飞速发展同样使计算机的运算速度愈来愈快。这样就使大量未知数的迭代方

程能快速完成，智能计算的大面积搜索能迅速实现。

（2）计算机的不足

① 计算机不能做随机变化的运算（计算机程序无法编制），只能按顺序、选择、循环的方式执行。

② 计算机不能对大数量的结点按指数增长的方式搜索（计算机运行时间太长，跟不上需求）。

### 3. 复杂问题的解决途径

复杂问题的求解需要把问题进行化解到计算机的本能所能解决的手段上来，即表示为

复杂问题求解＝计算机的本能＋问题化解后求解

### 4. 问题化解方法

（1）复杂问题的化解原则

① 所有复杂的数值计算问题都需要经过化简回归到“＋－×÷”。

② 所有复杂问题的运行结构都可用“顺序、选择、循环”三种基本结构的嵌套组合来完成。

③ 任何媒体数字化（二值化）后，就可以存入计算机并进行处理。

④ 充分利用计算机的大量存储空间和快速运算，把复杂的物体在空间上细化（如二值化表示、未知数结点增加），或使计算重复化（如迭代、搜索），即充分发挥计算机的优势。

（2）表达式的化解原则

表达式的化解原则是把人为的优先规定，变成前后顺序过程。

① 改变算术运算的“先乘除、后加减，括号优先”原则，成为“前后”顺序关系。

把算术表达式（中缀）变成逆波兰式（后缀）。这是编译原理中最关键的地方。

② 改变函数微分运算中，对表达式中求微分的顺序是先低级（＋、－）后高级（×、÷）的原则，成为顺序关系。

把表达式（中缀）变成波兰式（前缀），例如：

u× v+ p/q→+ × uv/pq

a× (b+ c)→× a+ bc

对任意的函数的中缀表达式变成前缀表达后，其导数求解时，每次就很自然地按前缀表达式的顺序套用微分公式，计算机就能顺利地求出此函数的导数。按这种方法编制的导数自动求解系统能作为高等数学课程的辅助答疑系统。

（3）采用间接运算方法

对于不能编制程序完成的随机求解的过程，可以采用间接运算方法，即：

随机求解过程＝间接运算＋循环运算

计算机程序中采用对数据地址的操作，开始了间接运算。这种方法可以扩展到对随机求解过程中去。

① 专家系统工具的研制

在知识库还是空时，只要规定知识的结构形式，就可以编制推理机程序。推理机程序是对知识结构进行操作，包括对知识的搜索、进栈、退栈、提问、解释等。编制推理机程序就是采用间接运算的方法。

② 遗传算法

遗传算法的重要特点是对个体编码位置的操作，而不是直接对编码本身的含义（参数）操作。这也是典型的间接运算。

③ 采用间接运算方法解决方程随机求解的问题

在求解运输问题的位势方程时，是一个是随机求解过程。例如，有如下位势方程：

$$c_1 + d_4 = 7,\quad c_2 + d_2 = 2,\quad c_2 + d_4 = 6$$

$$c_3 + d_1 = 9,\quad c_3 + d_3 = 4,\quad c_3 + d_4 = 8$$

以上 6 个方程 7 个未知数，在给定 $c_1=0$ 后，求解其他的 $c_i$ 和 $d_j$。这 6 个方程的求解顺序是跳跃式的。因为在方程中，只能在两个未知数中，有一个已求出、另一个未求出时，该方程才能求解。其他情况下都不能求解。对于这样的随机求解过程，程序是无法编制的。

为此，采用间接运算方法，即对每个未知数设计一个是否求出的标志位，顺序搜索每个方程，检查未知数的标志位是否符合求解要求，不符合时跳过该方程，符合时再检查未知数的标志位中哪个已求出（设为 1）、哪个未求出（设为 0），再求未求出的未知数的值。这样，把随机求解过程变成“间接求解加上循环顺序求解”过程。这种求解过程要循环多次才能完成。具体求解过程说明如下：

循环第一次位势方程的求解：第一个方程可求解（$c_1$ 标志位改为 1，$d_4$ 的标志位为 0），求得解为 $d_4=7$，$d_4$ 的标志位改为 1。第二个方程检查两个未知数的标志位均为 0，不能求解，跳过该方程。第三个方程检查两个未知数的标志位，$c_2$ 标志位为 0，$d_4$ 的标志位为 1，能求解，求得解为 $c_2=-1$，$c_2$ 标志位改为 1。第四个方程检查两个未知数的标志位均为 0，不能求解，跳过该方程。第五个方程检查两个未知数的标志位均为 0，不能求解，跳过该方程。第 6 个方程检查两个未知数的标志位，$c_3$ 标志位为 0，$d_4$ 的标志位为 1，能求解，求得解为 $c_3=1$。第一次循环结束，通过这次循环，求出的未知数 $d_4$、$c_2$、$c_3$ 的值。

按上方法再循环求解，经过多次循环求解就能够把其他未知数都求出解。这种间接运算方法把随机求解变成了间接求解加上循环顺序求解。

（4）有效使用标准程序工具

成熟的程序已经以工具的形式提供服务，如：

① 成熟的程序（如初等函数、绘图、数据库接口、网络应用等计算）作为标准的子程序放入子程序库中，通过连接并入应用程序中。

② 统计标准程序的工具，如 SAS、SPSS 等。

有效地使用标准程序工具将简化实际系统的编程。

（5）多资源组合形成解决问题的方案

决策资源有数据、模型、知识，有效地组合这些资源能达到辅助决策。组合的方法是编制一个总控制程序，通过调用这些资源的接口，按照程序的顺序、选择、循环的基本结构形式进行嵌套组合，形成多个方案，建立决策支持系统，用于解决决策问题。

### 5. 利用计算机的优势

(1) 扩大存储量

① 代数方程或微分方程的未知数已扩大到上万个。大面积的物理方程(天气预报等)的求解成为可能。

② 用点阵的二值数据表示汉字、声音、图像、视频等,开始了多媒体的处理(于 20 世纪 80 年代兴起)。

③ 数据库(二维数据)扩充为数据仓库(多维数据)。存放大量数据的数据仓库为辅助决策开辟了新方向(于 20 世纪 90 年代兴起)。

(2) 不惜计算时间

① 数值计算的迭代法

数值计算的迭代法就是不惜计算时间,进行重复计算,来求得方程的解,迭代次数可以是几万次或更多,只要是收敛的,就总能够得到满足精度的解。

② 用循环的顺序计算代替随机求解过程的计算

例如上面提到的运输问题的位势方程求解,它是随机求解过程,利用了间接运算加上循环运算进行求解,这是利用多化计算时间来代替随机求解过程的编程困难。

③ 知识推理中的知识搜索

在知识推理中对知识库中知识的多次反复搜索,完成了知识树的逆向推理。这也是不惜计算时间,简化了编程。

④ 人机博弈中走棋路径的搜索

人机博弈中,计算机的走步是计算对抗双方所有的棋子的走棋路径,通过棋局的静态估计函数,选择最佳走棋路径。这是典型的不惜计算时间,达到人难以思考的深度,即计算机计算对抗中,双方一人一步对抗的回合数能够多于人,从而战胜人。例如五子棋,计算机计算对抗双方一人一步的回合数,计算机可以搜索到最后的终止局面。人若犯一个错误,就将输给计算机。

若棋子多,双方对抗的回合次数又多,所有走棋路径将成指数次方的数量增长。要搜索所有走棋路径,一般需用亿次机来计算。

在国际象棋、中国象棋等的比赛中,计算机均战胜过人类高手。但是,围棋所有走棋路径按指数次方的数量增长,数量太大,计算机还无法完成。

### 6. 结束语

计算机(包括软件、硬件)虽然是非生物,但在人类的帮助下,计算机在模拟人的能力方面得到了飞速的发展。本书作者针对计算机进化过程进行了研究,发掘了一些进化规律,以便能更清楚地认识计算机的本质,这对于提高人们对计算机的使用效果,以及进一步促进计算机的进化起到了积极的作用。计算机进化规律的知识发现这个有意义的课题,希望能够唤起有兴趣者发掘更多的计算机进化规律,加速计算机的进化,使计算机更有效地为人类服务。

# 习　题　12

1. 数学变换在数学中起什么作用?

2. 变换规则知识与规则知识有什么不同? 为什么要研究变换规则知识?

3. 变换规则的知识挖掘定理1和定理2说明了什么问题?

4. 变换规则的知识挖掘过程是怎样的?

5. 变换规则的知识推理与一般规则的知识推理有什么不同?

6. 变换规则知识链定理说明了什么问题?

7. 在数据仓库中如何获取多种变换的变换规则知识链?

8. 用变换规则作为元知识的表示形式比一般规则作为元知识的表示形式有什么好处?

9. 神经网络的变换规则元知识是否说明了神经网络的本质?

10. 属性约简的变换规则元知识是否说明了属性约简的本质?

11. 专家系统的元知识采用变换规则元知识表示有什么好处?

12. 你理解的软件进化是什么?

13. 计算机的计算过程与人的计算过程有什么不同?

14. 计算机程序为什么采用对数据的存放地址的间接操作?

15. 为什么说"任何复杂的程序都是顺序、选择、循环这三个基本结构的嵌套组合",它能保证程序的正确性吗?

16. 数据库与数据仓库的数据有什么本质的不同? 它们是如何应用的?

17. 在知识推理中是如何进行大规模的知识搜索的?

18. 为什么说数值计算都要回到"加减乘除"?

19. 为什么说"比较"操作是逻辑计算的基础?

20. 为什么说汉字与多媒体要用二值数据来表示?

21. 你认为计算机的原始本能是什么?

22. 你认为复杂问题的求解需要把问题化解到计算机的本能所能解决的手段上来吗?

23. 对于随机求解过程的问题,计算机编程采用了哪些方法?

24. 用间接求解方法能否代替"把表达式变化为逆波兰式"方法?

25. 你也来总结一下软件进化的规律,共同提高大家的认识。

26. 你认为计算机有什么不足? 如何来克服,并进一步促进计算机的进化?

# 第13章 文本挖掘与Web挖掘

## 13.1 文本挖掘概述

### 13.1.1 文本挖掘的基本概念

在现实世界中，人们面对的数据大都是文本数据，由各种数据源(如新闻文章、研究论文、书籍、数字图书馆、电子邮件和 Web 页面)的大量文本组成。由于文本的信息量的飞速增长，如电子出版物、电子邮件、CD-ROM 和 Web 等。Web 中 99%的可分析信息是以文本形式存在的。Web 网页总量已达数百亿，每天新增网页数千万，截至 2008 年年底，中国网页总数超过 160 亿个。

文本数据是半结构化数据，它既不是完全无结构的，也不是完全结构化的。例如，文本可能包含结构字段，如标题、作者、出版日期、长度、分类等，也可能包含大量的非结构化的文本，如摘要和内容。

文本挖掘与数据挖掘的区别在于：数据挖掘的对象以数据库中的结构化数据为主，并利用关系表等存储结构来发现知识。文本挖掘中文档本身是半结构化的或非结构化的，无确定形式，并且缺乏机器可理解的语义。因此，数据挖掘的技术不完全适用于文本挖掘，至少需要进行预处理。

文本挖掘(Text Mining)，也称为文本数据挖掘(Text Data Mining)。文本挖掘一词出现于 1998 年第十届欧洲机器学习会议(the European Conference On Machine Learning, ECML'98)上，首次进行了关于文本挖掘的专题讨论会。组织者 Kodratoff 明确地定义了文本挖掘的概念，他认为文本挖掘的目的是从文本集合中搜寻知识，即在目前对自然语言理解的水平上，利用该领域的成果，试图尽可能多地提取知识。因此，文本挖掘需要数据挖掘、语言学、数据库以及文本标记和理解方面的专家的参与。

**1. 概念**

文本挖掘是一个从大量文本数据中提取以前未知的、有用的、可理解的、可操作的知识的过程。文本数据包括技术报告、文本集、新闻、电子邮件、网页、用户手册等。文本挖掘对单个文本或文本集(如 Web 搜索中返回的结果集)进行分析，从中提取概念，并按照指定的方案组织、概括文本，发现文本集中重要的主题。它除了从文本中提取关键词外，还要提取事实、作者的意图、期望和主张等。这些知识对许多应用目标，如市场营销、趋势分析、需求处理等，都是很有用的。

**2. 主要任务**

文本挖掘的任务主要是：

(1) 短语和特征的提取。在读取大量的非结构化文本时，应用自然语言处理技术提取文本集中所有相关的短语。文本内容可看成由它所包含的基本语言单位(字、词、词组或短语等)组成的集合。在短语提取中，还要将非结构化的原始文本短语集合的内容转换为更加容易处理的概念级数据。可以形象地把文本挖掘看做是一支荧光笔，它通读文本时高亮度显示有关的短语，这些短语放在一起就可以得到对文本的一个较好的概括性理解。

对于能够描述和说明文本的短语，可称之为文本的特征。短语和特征的提取是文本挖掘的首要任务。

(2) 文本关联分析。文本挖掘的核心功能表现为分析一个文本集合中的各个文本之间概念共同出现的模式。实际上，文本挖掘依靠算法和启发式方法，跨文本考虑概念分布、频繁概念(项)以及各种概念的关联，其目的是使用户发现概念的关联，这种概念的关联是文本集合作为一个整体所反映出来的。

(3) 文本聚类与文本分类。文本聚类是对文本集合中的各个文本之间从没有类别，按就近原则聚合成类。文本分类是对文本集合中已经有了类，建立起各类的规则知识，按此规则对新文本进行分类。

#### 3. 文本挖掘与数据挖掘

文本挖掘与数据挖掘相比，它们的相似点在于两者都处理大量的数据，都可归属到知识发现领域中。它们之间的差别在于许多经典的数据挖掘算法，如数值预测、决策树等都不太适用于文本挖掘，因为它们依赖于结构化的数据，而短语或概念关联分析等工作则是文本挖掘所独有的，如表 13.1 所示。

**表 13.1 文本挖掘与数据挖掘的区别**

| | 数 据 挖 掘 | 文 本 挖 掘 |
|---|---|---|
| 研究对象 | 用数字表示的、结构化的数据 | 无结构或者半结构化的文本 |
| 对象结构 | 关系数据库 | 自由开放的文本 |
| 目标 | 获取知识，预测以后的状态 | 提取概念和知识 |
| 方法 | 归纳学习、决策树、神经网络、粗糙集、遗传算法等 | 提取短语、形成概念、关联分析、聚类、分类 |
| 成熟度 | 从 1994 年开始得到广泛应用 | 从 2000 年开始得到广泛应用 |

### 13.1.2 文本特征的表示

与数据库中的结构化数据相比，文本具有有限的结构，或者说根本就没有结构。即使具有一些结构，也是着重于格式，而非文本内容。不同类型文本的结构也不一致。此外，文本的内容是人类所使用的自然语言，计算机很难处理其语义。文本信息源的这些特殊性使得现有的数据挖掘技术无法直接应用于其上，需要对文本进行预处理，抽取代表其特征的元数据。这些特征可以用结构化的形式保存，作为文本的中间表示形式。

文本特征指的是关于文本的元数据，分为：①描述性特征，例如文本的名称、日期、大

小、类型等；②语义性特征，例如文本的作者、机构、标题、内容等。描述性特征易于获得，而语义性特征则较难得到。对于内容这个难以表示的特征，我们首先要找到一种能够被计算机所处理的表示方法。

矢量空间模型（VSM）是效果较好的表示文本特征的方法。在该模型中，文本空间被看做是由一组正交词条矢量所形成的矢量空间，每个文本 $d$ 表示为其中的一个规范化特征矢量：

$$V(d)=(t_1,w_1(d);\cdots;t_i,w_i(d);\cdots;t_n,w_n(d))$$

其中 $t_i$ 为词条项，$w_i(d)$ 为 $t_i$ 在 $d$ 中的权值。可以将 $d$ 中出现的所有单词或所有短语作为 $t_i$，从而提高内容特征表示的准确性。$w_i(d)$ 一般被定义为 $t_i$ 在 $d$ 中出现频率 $tf_i(d)$ 的函数，即 $w_i(d)=\Psi(tf_i(d))$。常用的 $\Psi$ 有：

（1）布尔函数

$$\Psi=\begin{cases}1, & tf_i(d)>0\\ 0, & tf_i(d)=0\end{cases}$$

（2）平方根函数

$$\Psi=\sqrt{tf_i(d)}$$

（3）对数函数

$$\Psi=\log(tf_i(d)+1)$$

### 13.1.3　文本特征的提取

特征提取主要是识别文本中代表其特征的词项。提取过程是自动的，提取的特征大部分是文本集中表示的概念。文本特征分为一般特征和数字特征，其中一般特征主要包括动词和名词短语，如人名、组织名等；数字特征主要包括日期、时间、货币以及单纯数字信息。这些特征包含重要的信息，因此特征提取是一种强有力的文本挖掘技术。通过文本特征抽取，用于记录文本的特征，可以更好地组织文本，如文本的存储、检索、过滤、分类和摘要等。

中文姓名识别属于中文信息处理中未登录词处理的范畴，中文姓名在文章中的出现频率虽然不高，但绝不可以忽略，因为中文姓名本身包含着重要的信息，它可能是整个句子甚至整个段落的语义中心，如果不予处理，将影响文本挖掘的性能。数字特征反映一定的信息，但不能表达文本的中心思想，通常只作为文本挖掘中的参考信息。姓名特征提取算法所提取的姓名特征，作为文本内容的特征表示。

构成文本的词汇，数量是相当大的，因此，表示文本的向量空间的维数也相当大，可以达到几万维，因此需要压缩维数，这样做的目的主要有两个，第一，提高程序的效率，提高运行速度，第二，所有几万个词汇对文本分类的意义是不同的，一些通用的、各个类别都普遍存在的词汇对分类的贡献小，在某特定类中出现比重大而在其他类中出现比重小的词汇对文本分类的贡献大。

为了提高分类精度，对于每一类，都应去除那些表现力不强的词汇，筛选出针对该类的特征项集合。目前存在多种筛选特征项的算法，如根据词和类别的互信息量判断、根据词熵判断等。

例如，根据词和类别的互信息量进行特征项（能体现类别的词）抽取的判断算法过程

如下：

（1）初始情况下，该特征项集合包含所有该类中出现的词。

（2）对于每个词，计算词 $W_i$ 和类别 $C_j$ 的互信息量 $I(W,C)$。

（3）对于该类中所有的词，依据上面计算的互信息量排序。

（4）抽取一定数量的词（互信息量大的词）作为特征项，具体需要抽取多少特征项，目前尚无很好的解决方法，一般采用先定初始值，然后根据实验测试和统计结果确定最佳值，一般初始值定在几千左右。

（5）将每类中所有的训练文本，根据抽取的特征项，进行向量压缩，精简向量表示。

# 13.2 文本挖掘

## 13.2.1 文本挖掘功能层次

文本挖掘的功能可以用一个层次结构表示，如图 13.1 所示。

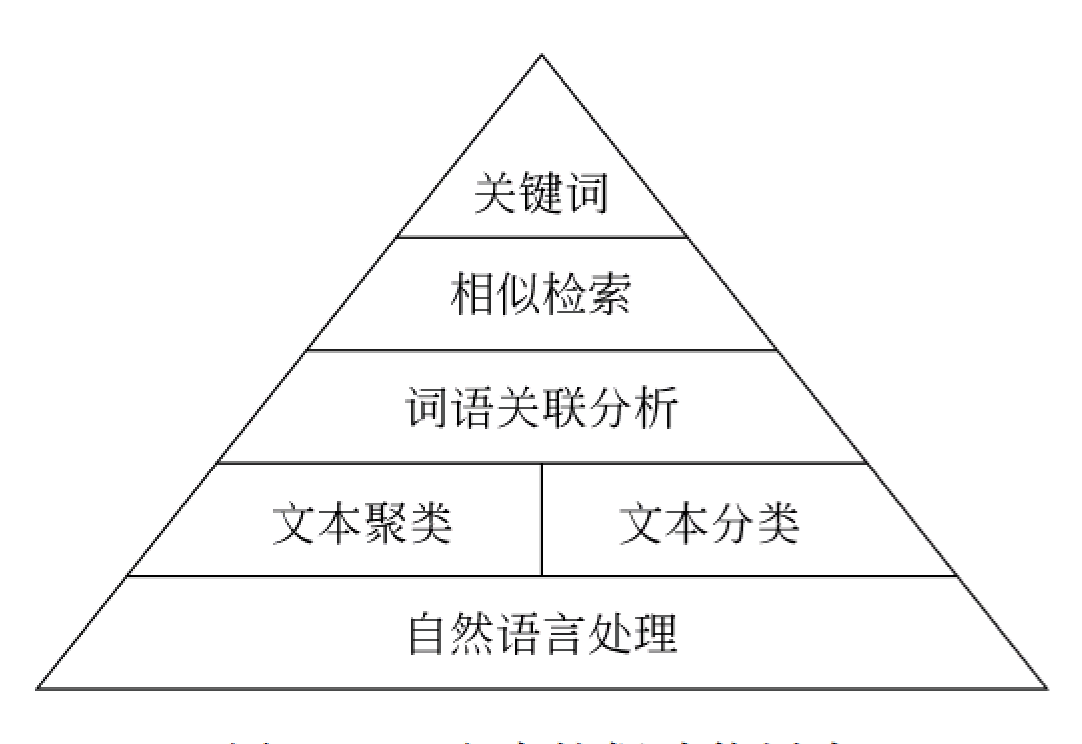



图 13.1 文本挖掘功能层次

文本挖掘功能从顶端到底端说明如下。

**1. 关键词检索**

关键词建立倒排文件索引。简单的搜索引擎通常基于关键词检索相关文档，该技术与传统的信息检索使用的技术类似。

**2. 相似检索**

它与信息检索方法中的相似性检索方法类似，目的是找到相似内容的文本。

**3. 词语关联分析**

它不仅将注意力放在孤立的词语的相同或相似信息上，而且聚焦在词语（包括关键词）之间的关联信息分析上。从而避免了传统的信息检索技术带来的信息不精确和信息量过大等问题。

**4. 文本聚类和文本分类**

利用类似于数据挖掘的聚类和分类技术实现文本的聚类和分类，将文本在一个更高层

次上进行抽象和整理。

**5. 自然语言处理**

这是最复杂的功能,它希望揭示自然语言处理技术的语义,进行文本语义挖掘。

目前文本挖掘主要是词语关联分析、文本聚类和文本分类工作。

### 13.2.2 文本关联分析

基于关键词或短语的关联分析首先收集经常一起出现的关键词或短语,然后找出其关联或相互关系。

关联分析首先要对文本数据进行词根处理,去除非用词等预处理,然后调用关联挖掘算法。在文本数据库中,每一文本被视为一个事务,文本中的关键词组可视为事务中的一组事务项,即文本数据库可表示为:

{文本编号,关键词集}

文本数据库中关键词关联挖掘的问题就变成事务数据库中事务项的关联挖掘。

注意一组经常连续出现或紧密相关的关键词可形成一个词或词组。关联挖掘有助于找出复合关联(compound association),即领域相关的词或词组,如[科技大学,大学]或[总统,克林顿],或非复合关联,如[美元,参股,交易,总额,佣金,赌注,证券]。基于这些词关联的挖掘称为“词级(term level)关联挖掘”。

词的识别和词组关联挖掘在文本分析中有两个优点:①词和词组被自动标记,无需人去标记文本;②挖掘算法的执行时间和无意义的结果将极大地减少。

利用这种词和词组的识别,关联分析挖掘可以用于找出词或关键词间的关联。一些用户可能喜欢从给定关键词或词组中找出关键词或词组之间的关联,而有些用户可能希望找出一起出现的最大词集。因此,根据用户挖掘的需要,可以使用关联挖掘或最大模式挖掘算法。

### 13.2.3 文本聚类

文本聚类是一种典型的无教师的机器学习问题。目前的文本聚类方法大致可以分为层次聚类法和平面划分法两种类型。

**1. 层次聚类法**

对于给定的文本集合 $D=\{d_1,\cdots,d_i,\cdots,d_n\}$,层次聚类法的具体过程如下:

(1) 将 $D$ 中的每个文本 $d_i$ 看作是一个具有单成员的类 $c_i=\{d_i\}$,这些类构成了 $D$ 的一个聚类 $C=\{c_1,\cdots,c_i,\cdots,c_n\}$;

(2) 计算 $C$ 中每对类 $(c_i,c_j)$ 之间的相似度 $\mathrm{sim}(c_i,c_j)$;

(3) 选取具有最大相似度的类对 $\max\limits_{c_i,c_j\in C}\mathrm{sim}(c_i,c_j)$,并将 $c_i$ 和 $c_j$ 合并为一个新的类 $c_k=c_i\cup c_j$,从而构成了 $D$ 的一个新的聚类 $C=\{c_1,\cdots,c_{n-1}\}$;

(4) 重复上述步骤,直至 $C$ 中剩下一个类为止。

该过程构造出一棵生成树，其中包含了类的层次信息，以及所有“类”内和“类”间的相似度。层次聚类法是最为常用的聚类方法，它能够生成层次化的嵌套类，且准确度较高。但是，在每次合并时，需要全局地比较所有类之间的相似度，并选择出最佳的两个类，因此运行速度较慢，不适合于大量文本的集合。

**2. 平面划分法**

平面划分法与层次聚类法的区别在于，它将文本集合水平地分割为若干个类，而不是生成层次化的嵌套类。对于给定的文本集合 $D=\{d_1,\cdots,d_i,\cdots,d_n\}$，平面划分法的具体过程如下：

(1) 确定要生成的类的数目 $k$；

(2) 按照某种原则生成 $k$ 个聚类中心作为聚类的种子 $S=\{s_1,\cdots,s_j,\cdots,s_k\}$；

(3) 对 $D$ 中的每个文本 $d_i$，依次计算它与各个种子 $s_j$ 的相似度 $\mathrm{sim}(d_i,s_j)$；

(4) 选取具有最大相似度的种子 $\max\limits_{s_j\in S}\mathrm{sim}(d_i,s_j)$，将 $d_i$ 归入以 $s_j$ 为聚类中心的类 $c_j$，从而得到 $D$ 的一个聚类 $C=\{c_1,\cdots,c_k\}$；

(5) 重复步骤(2)、(3)、(4)若干次，以得到较为稳定的聚类结果。该方法的运行速度较快，但是必须事先确定 $k$ 的取值，且种子选取的好坏对聚类结果有较大影响。

### 13.2.4 文本分类

文本分类是一种重要的文本挖掘工作，由于现在存在大量的联机文本，分类便于对文本的检索和分析。

“如何进行自动文本分类?”一般的做法如下：首先，把一组预先聚类过的文本作为训练集。然后对训练集进行分析以便得出各类的分类模式。这种分类模式通常需要一定的测试过程，不断地细化，用这些导出的分类模式对其他联机文本加以分类。

这一处理过程与关系数据库的分类相似，但还是存在本质的区别。关系数据库是结构化的：每个元组定义为一组“属性，值”对。文本数据库则不是结构化的，它没有“属性，值”对的结构。与一组文本相关的关键词并不能用一组属性或维来刻化。因此，通常面对关系数据库的分类方法，如决策树分析，并不适用于对文本数据库的分类。

对文本分类的有效方法是基于关联的分类，它基于一组关联的、经常出现的文本模式对文本加以分类。基于关联的分类方法处理过程如下：

(1) 通过简单的信息检索技术和关联分析技术提出关键词和词组。

(2) 使用已经有的词类，或基于专家知识，或使用某些关键词分类方法，生成关键词和词组的概念层次，或类层次结构。

(3) 词关联挖掘方法用于发现关联词，它可以最大化区分一类文本与另一类文本。这导致了对每一类文本，有一组关联规则。这些分类规则可以基于其出现频率加以排序，并用于对新的文本的分类。

基于关联的文本分类方法已经证明是有效的。对 Web 文本分类，可以利用 Web 页面的链接信息，帮助文本类的识别。

文本分类是一种典型的有教师的机器学习问题，一般分为训练和分类两个阶段，具体过

程如下：

**1. 训练阶段**

(1) 定义类别集合 $C=\{c_1,\cdots,c_i,\cdots,c_m\}$，这些类别可以是层次式的，也可以是并列式的；

(2) 给出训练文本集合 $S=\{s_1,\cdots,s_j,\cdots,s_n\}$，每个训练文本 $s_j$ 被标上所属的类别标识 $c_i$；

(3) 统计 $S$ 中所有文本的特征矢量 $V(s_j)$，确定代表 $C$ 中每个类别的特征矢量 $V(c_i)$。

**2. 分类阶段**

(1) 对于测试文本集合 $T=\{d_1,\cdots,d_k,\cdots,d_r\}$ 中的每个待分类文本 $d_k$，计算其特征矢量 $V(d_k)$ 与每个 $V(c_i)$ 之间的相似度 $\mathrm{sim}(d_k,c_i)$；

(2) 选取相似度最大的一个类别 $\max\limits_{c_i\in C}\mathrm{sim}(d_k,c_i)$ 作为 $d_k$ 的类别。

有时也可以为 $d_k$ 指定多个类别，只要 $d_k$ 与这些类别之间的相似度超过某个预定的阈值。如果 $d_k$ 与所有类别的相似度均低于阈值，那么通常将该文本放在一边，由用户来做最终决定。如果这种情况经常发生，则说明需要修改预定义类别，然后重新进行上述训练与分类过程。在计算 $\mathrm{sim}(d_k,c_i)$ 时，有多种方法可供选择。最简单的方法是仅考虑两个特征矢量中所包含的词条的重叠程度，即：

$$\mathrm{sim}(d_k,c_i)=\frac{n(d_k,c_i)}{n_0(d_k,c_i)}$$

其中，$n(d_k,c_i)$ 是 $V(d_k)$ 和 $V(c_i)$ 具有的相同词条数目，$n_0(d_k,c_i)$ 是 $V(d_k)$ 和 $V(c_i)$ 具有的所有词条数目。最常用的方法是考虑两个特征矢量之间的夹角余弦。

## 13.3 Web 挖掘

万维网(WWW)目前是一个巨大的、分布广泛的和全球性的信息服务中心，它涉及新闻、广告、消费信息、金融管理、教育、政府、电子商务和许多其他信息服务。Web 还包含了丰富和动态的超链接信息，以及 Web 页面的访问和使用信息，这为数据挖掘提供了丰富的资源。从广义上讲，Web 信息也是一类特别的文本信息，因此文本挖掘的各种技术也适合于 Web 挖掘，但是由于 Web 信息自身的特点，对于文本挖掘和 Web 挖掘应该区别对待。

### 13.3.1 Web 挖掘概述

**1. Web 信息特点**

(1) Web 信息特别庞大

Web 的数据量目前以几百兆兆字节为单位来计算，而且仍然在迅速地增长。许多机构和社团都在把各自大量的可访问信息置于网上。

(2) Web 信息非常复杂

Web 可以看做一个巨大的数字图书馆;然而,这一图书馆中的大量文本并不根据任何有关排列次序加以组织。它没有分类索引,更没有按标题、作者、封面页、目录等的索引。在这样一个图书馆中搜索希望得到的信息是极具挑战性的。

(3) Web 信息是动态的

Web 不仅以极快的速度增长,而且其信息还在不断地发生着更新。新闻、股票市场、公司广告和 Web 服务中心都在不断地更新着各自的页面。链接信息和访问记录也处在频繁地更新之中。

(4) Web 信息使用者复杂

Web 面对的是一个广泛的形形色色的用户群体。目前因特网上连接有约 5000 万台工作站,其用户群仍在不断地扩展中。各个用户可以有不同的背景、兴趣和使用目的。

(5) Web 信息中的"垃圾"非常多

一个人只是关心 Web 上很小的一部分信息,Web 所包含的其余信息,用户是不感兴趣的,而且会淹没用户所希望得到的搜索结果。

### 2. Web 数据挖掘的意义

Web 挖掘的实质就是从 Web 页面及其链接和用户对页面的访问中挖掘出用户感兴趣的知识。通过 Web 数据挖掘,可以从数以亿计存储大量多种多样信息的 Web 页面及其链接和用户对页面的访问中挖掘出需要的有用知识。

数据挖掘使得商家能更好地了解客户,同时也使得经济规模达到价格更低廉和选择更多,使一个好企业更好。例如,在描述大量客户意见的信息中,通过数据挖掘来提出一个模型,该模型具备一个通用的区分客户抱怨还是赞扬的能力。客户的抱怨可以给公司一个机会,让公司学到怎样改进策略以使将来不满意的客户越来越少。

Web 挖掘的数据来源是网站数据,这些数据包括网页文本信息、网页链接信息、网站的访问记录以及其他可收集的信息。但是,不同的挖掘目的、不同的挖掘算法总是依靠不同的一种或几种数据源,例如 Web 日志(服务器日志、错误日志、Cookie 日志等)、在线市场数据、Web 页面、Web 页面超链接以及包括用户注册信息等数据源。

### 3. Web 挖掘分类

可以将 Web 挖掘一般地定义为:从 WWW 的资源和行为中抽取感兴趣的、有用的模式和隐含的信息。一般地,Web 挖掘可分为 3 类:Web 内容挖掘(Web content mining)、Web 结构挖掘(Web structure mining)和 Web 应用挖掘(Web usage mining)。

图 13.2 为 Web 挖掘的分类图。

(1) Web 内容挖掘

内容挖掘是用来提取文字、图片或其他组成网页内容成分的信息和知识。哪个站点卖汽车?哪些页面是中文的?哪些页面是介绍音乐的,或是介绍新闻的?搜索引擎、智能代理和一些推荐引擎都使用内容挖掘来帮助客户在浩瀚的网络空间中寻找所需的内容。

Web 内容挖掘有两种策略:①页面文本内容挖掘;②对搜索引擎的查询结果进行进一

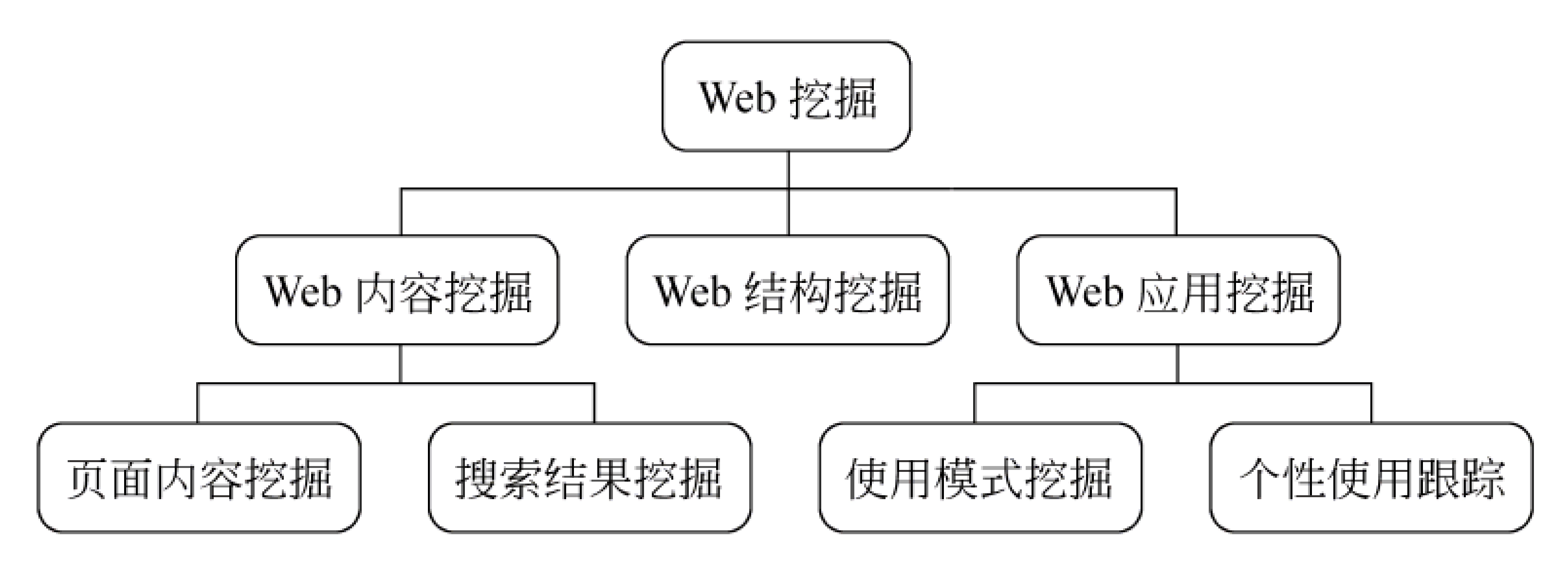



图 13.2　Web 挖掘分类图

步的处理，得到更为精确和有用的信息。

(2) Web 结构挖掘

结构挖掘是用来提取网络的拓扑信息，即网页之间的链接信息。从 WWW 的组织结构和链接关系中挖掘知识。哪些页面被其他页面所链接？哪些页面指向了其他页面？哪些页面的集合构成了一个独立的整体？可以对页面进行排序，发现重要的页面。

(3) Web 应用(访问信息)挖掘

应用挖掘是用来提取关于客户如何运用浏览器浏览和使用页面链接的信息。从 Web 的访问记录中抽取感兴趣的模式。客户访问了哪些页面？在每一页上待了多长时间？下一步点击了什么？在站点中是按照怎样的访问路线进入和退出的？

WWW 中的每个服务器都保留了访问日志(Web access log)，记录了关于用户访问和交互的信息。分析这些数据可以帮助理解用户的行为，从而改进站点的结构，或为用户提供个性化的服务。

这方面的研究主要有两个方向：一般使用模式的挖掘和个性化使用记录的追踪。一般使用模式的挖掘通过分析使用记录来了解用户的使用模式和倾向，以改进站点的组织结构；而个性化使用记录的追踪则倾向于分析单个用户的偏好，其目的是根据不同用户的访问模式，为每个用户提供定制的站点。

(4) 区别与联系

因特网是由许多用链接联系起来的网页组成的。每个单独的页面都由多种成分组成，例如文本、图片及指向其他页面的链接等；网络服务器提供了对这些成分的访问权限。一个网页是由一些称为框架(frame)的结构组成的。

进行结构挖掘的原材料是一套将文档联系起来的超级链接。内容挖掘的原材料由那些存储于数以百万的文件中的文本组成，这些文件可以让任何客户通过网络浏览器来访问。内容挖掘和结构挖掘都需要一种相对的静态的网络，也就是说，网页和链接要像静止在某个特定的时刻。

对结构挖掘的理想的表达方式是用图形的方式(实际上是有向图，因为链接总是在一个方向上由一个网页指向另一个)。这种理想的图可以映射整个网络中链接所有文档的全部链接，像一个索引，这个理想化的索引链接网络上每个网页中的每一个字符串、单词、短语、声音和图像。

结构挖掘提示了哪些页面通过当前页可以在几步内到达，但并不关心多少人会实际用到这条通路。内容挖掘提示了网页的主题，但并不关心谁会真正地阅读它。内容挖掘可以

用于找出所有关于酒类的网页，而结构挖掘可以将这些网页组织成零售站点的聚类。

对于葡萄酒的购买者和白酒的购买者的区别，就需要另一种类型的Web挖掘，即第三种称为应用(访问信息)挖掘的Web挖掘，它主要集中于挖掘客户的行为，特别是随着时间的变化。有时感兴趣的时间片很短，例如对于访问者在一次单独的会话中在一个站点中的访问路径的分析；在其他时候时间片又会比较长，例如对于在一个零售站点长期注册的购买者的购买行为的分析。

应用(访问信息)挖掘的理想的数据表现形式是客户应用模式，它记录或描述了某个单独的客户与网络的交互情况，包括所访问的站点，访问的路线，提出的问题，阅读的文档和购买的物品等。

结构挖掘、应用挖掘和内容挖掘都是Web挖掘的有价值的应用，它们完全都可以被称为"对网络的挖掘"。

### 13.3.2　Web内容挖掘

内容挖掘是从组成WWW的网页中提取信息的过程。内容挖掘最广为人知的一个应用是搜索引擎。没有它，网络将变得一无是处。Web内容挖掘的基本技术是文本挖掘。

#### 1. 信息检索

网络上有数不清的信息、留言，还有彻头彻尾的垃圾。找到需要的信息是一件不太容易的搜索工作，因为对于大多数的主题来说，网络只是一个"贫矿"。如果在网络上的所有文件都被明确地标记了关键词或是可以清楚描述文章内容的元数据，客户就可以向图书馆管理员那样使用搜索引擎。那样搜索起来就不需要那么复杂的算法，只要简单地查询就行了。

信息检索的目标是找到用户想要找的，而不理会其他。这个想法可以由研究者从两个方面来判断该查询的有效性："召回(recall)"和"精度(precision)"。"精度"回答了"在返回的网页中，正确的标题的比例是多少"的问题；"召回"则是回答"返回了多少"正确网页的问题。这两个目标在某种程度上说是矛盾的。一个搜索引擎针对任何一个请求返回所有的网页可以说有了很高的"召回"，但是只有很低的"精度"；反之，只返回一个正确主题网页的搜索引擎可以说有很高的"精度"，但"召回"很低。

"召回"和"精度"哪个更重要？那要看查询的性质。一些问题可以在查找到的一个网页里轻易回答，有些则要参照很多网页。

搜索引擎努力地提高精度和召回数量，这两者都依靠于按主题分类的能力——这也是数据挖掘中一个十分吸引人的挑战。

#### 2. 从纯文本中提取信息

内容挖掘的目的就是从纯文本中得到有用的信息，即基于页面内容相似度进行用户分类或聚类的，通过用户过去的检索内容分析完成个性化的建立。要达到这样的程度，就必须真正地理解文本，而这样的程度还没有达到。但是在一个有限制的范围内，识别出一些特定的信息是可能的。那些追求信息提取的研究者的一个希望就是通过将纯文本转化为结构化的数据，他们能够直接应用数据挖掘技术从而做出预测。这种从非结构化数据中创建结构

化数据的过程叫做特征抽取。例如,政府研究机构希望使用该技术来扫描如“泵设计”的领域中数量巨大的页面,以寻找诸如泵的容量和操作的压力,而这一切的目的是预测这一地区水的流量。

特征抽取在网络上的应用是作为购物的一个辅助工具。有许多这样的服务,这些服务是寻找电子商务的站点并比较相同商品的价格。这要求它要有识别出两个站点正在销售同一商品(网站在卖着某些东西)的能力。

### 13.3.3 Web 结构挖掘

结构挖掘可以告诉用户一些站点的受欢迎程度和它同其他站点的距离(通过跳转次数来判定)。深入一步,还可以通过查看一个单独站点的网页的链接情况及相互链接的情况来学习其内部结构。

网络的总体结构是十分迷人的。一个对于网络的分析将提示出人类分为数个不同的语言群落,并且任何以某种语言写成的页面总是链接与它相同语言的页面。

万维网(www)是一个有向图 $G=(V,E)$,$V$ 是页面的集合,$E$ 是页面之间的超链接集合。页面抽象为图中的顶点,而页面之间的超链接抽象为图中的有向边。顶点 $v$ 的入边表示对 $v$ 的引用,出边表示 $v$ 引用了其他的页面。所以 Web 页面之间的超链接揭示了 Web 结构。

每个网页是这个图的一个结点,每个链接是一条边。之所以说这个图是有向的,是因为存在由 $A$ 指向 $B$ 的链接并不等于也存在 $B$ 指向 $A$ 的对应链接。一个站点 $A$,它的每一个网页都包含了一个指向主页的链接。大部分的链接都是站内的,也可以指向站外的网页。

#### 1. 网页的引用

在“不是出版,就是毁灭”的学术世界里,引用一直是保持成绩的一个方法。仅仅是出版过文章是不够的,重要的是其他人的确读过它们并且觉得它们有用。一篇文章的有用与否在于这篇文章出现在其他文章的参考书目中的次数。特别是作者,会因为其作品的重复引用而在某个学科领域出名。

原则上讲,网络这种全球性结构也以同样的方式使网站保持成绩。通向这个站点链接越多,它就一定越重要。实际上对于站点管理者来讲,得到一个关于所有链接的准确视图是非常困难的,因为网络的结构绝不是静态的。被各大搜索站点用于建立索引的“网络爬行者”(Web crawler)是最易得到这种信息的来源。

网页引用的 Page-rank 方法是 Brin 和 Page 于 1998 年提出的一种方法。假设要搜索某一给定话题的 Web 页面,例如金融投资方面的页面。这时除了希望得到与之相关的 Web 页面外,还希望所检索到的页面具有较高质量和权威性。权威性(authority)可由 Web 页面链接来反映。Web 不仅由页面组成,而且还包含了从一个页面指向另一个页面的超链接。超链接包含了大量人类潜在的语义,它有助于自动分析出权威性语义。当一个 Web 页面的作者建立指向另一个页面的指针时,可以看做是作者对另一页面的注解。把对一个页面的来自不同作者的注解收集起来,就可以用来反映该页面的重要性,并可以很自然地用于 Web 页面权威性的发现。可见,大量的 Web 链接信息提供了丰富的关于 Web 内容相关性、

质量和结构方面的信息，这对 Web 挖掘是可以利用的一个重要资源。

Page-rank 的基本思想是：

(1) 一个页面被多次引用，则这个页面很可能是重要的；

(2) 一个页面尽管没有被多次引用，但被一个重要页面引用，则这个页面很可能是重要的；

(3) 一个页面的重要性被均分并被传递到它所引用的页面。

### 2. 中枢和权威

要在庞大的满足条件的文档中找到最有趣的或最权威的文档是非常困难的。

康奈尔大学的 Jon Kleinberg 提出了一种被广泛采用的技术来解决这个问题。他的想法是利用这样的事实：在建立从一个站点到另一个站点的链接时，网站的管理者将会对将要建立链接的网站的价值做一个判断。每个到站点的链接对这个站点都是有意义的。久而久之，那些决定给同一目标站点提供链接的站点能够证实目标的权威性。进一步，所要链接的站点的可靠性也可以通过它们链接到的站点的权威性来判断。一个拥有许多其他好站点推荐的站点的推荐可以用来决定另一个站点的权威性。

Kleinberg 提出一个链接到许多权威站点的站点叫做中枢(hub)；被许多中枢链接的站点叫做权威(authority)。这两个概念放在一起可以辨别出权威和大众化站点(如 Yahoo)之间的区别。一种寻找权威的结构化的方法就是，用其他的站点到该站点的链接数来将它们分级。要给站点分级，不要用指向它们的链接的总数，而是用指向它们标题相关的中枢的数量来分级。

结构挖掘是为提取信息而对网站的链接进行分析的过程，对单一网站局部结构的分析，对于理解此网站的创办的目的和设计很有帮助。对全局结构的分析是一种将一个网站分解成多个紧密联系的子网站的途径。运用全局结构挖掘，有可能把网页归类为中枢(到许多其他网页的很好的跳板网页)和权威(许多网页设计师都觉得值得链接到的网页)。

### 3. 导航页

导航页的存在主要为了链接其他页面。客户不必在导航页上花费太多的时间，而他们却会频繁地转到这个页面上。对客户来说，导航页使他们能够很容易地找到他们想要找的网页。通过比较从入口到目标网页所要求的点击数和浏览者平均的点击数，就会得到一些关于怎样设计好的网络站点和怎样链接网页的建议。

### 4. 目标页

浏览者通常花费大量的时间在目标页上。这一网页实际上给浏览者提供所要查找的信息、娱乐和商品——简而言之，目标页给浏览者提供所有的内容。

目标页一般是固定的。当浏览者在一个目标页上花费了大量的时间时，我们希望这是因为他们找到了他们所需要的东西。当然，并不是所有的浏览者都是这样的。或许他们有许多的疑惑，或者要求查到更多的东西，要么由于其他的原因使他们的输入速度非常慢，从而导致了他们在此网页上花费了大量的时间。通常仔细分析登录数据，就可以得出他们的

不同之处。要指出的重要的一点是：如果没有应用数据的配合，一个网站的静态结构是没有太大用处的。应用数据允许我们比较这个网站的结构，因为它反映了设计者的思想，也就是说反映的是这个网站及其实际的行为数据该如何使用。

### 13.3.4　Web 应用（访问信息）挖掘

在 Web 应用挖掘中，考虑的是对客户的理解，这时客户应用模式就是非常关键的。客户应用模式可以从多个层次检测和挖掘到，即从单个客户在一次对话中的一系列的单击到跨越了几个月或数年的客户群的购买中获取应用模式。

应用挖掘有很多应用，从提高网站的设计到改善客户关系的管理。随着人们需求的不断增长，所要求的数据资源也更加丰富多变。

#### 1. Web 应用挖掘的意义

Web 应用挖掘的意义可以概括为如下几点。

(1) 改进 Web 站点的效率。通过对用户访问信息的挖掘，得到大多数用户的访问习惯、爱好和其他有用信息，利用这些信息可以指导网站提供商改进站点结构和布局，吸引更多用户。

(2) 实现个性化服务。随着互联网的普及和电子商务的发展，电子商务系统在为用户提供越来越多选择的同时，其结构也变得更加复杂，用户经常会迷失在大量的商品信息空间中，无法顺利找到自己需要的商品。在日趋激烈的竞争环境下，个性化服务是包括电子商务在内的网站提供商争取更多用户、防止用户流失以及实现市场目标的重要手段。

(3) 商业知识的发现。从过去的访问信息特性的挖掘，发现新的商业知识，用于指导改进服务和扩展新的赢利点。通过结合日志数据和市场数据可以和 CRM 管理结合，在诸如：顾客吸引(Customer Attraction)、顾客保留(Customer Retention)、跨区销售(Cross Sales)、顾客离开(Customer Departure)等市场活动中，利用商业知识找到相应的最佳对策。

(4) 发现导航模式。用户的导航模式是指群体用户对 Web 站点内的页面的浏览顺序模式。在电子商务环境下发现商业知识的关键是发现用户的导航模式。这种导航模式也是个性化推销的基础。

(5) 抽取访问信息特性。通过对客户端，服务器端，代理服务器端等不同用户访问信息的挖掘，可以得到关于用户交互情况和导航情况的详细的信息。在此基础上可以提出模型，用于预测在一个给定站点上一个用户所访问的页面的概率分布。访问信息的特性可以被用于在 Web 服务器上开展伸缩性和负载均衡的研究等方面。

#### 2. Web 应用挖掘中的技术

Web 应用挖掘中的常用技术有如下几种。

(1) 路径分析。路径分析最常见的应用是用于判定在一个 Web 站点中最频繁访问的路径，这样的知识对于一个电子商务网站或者信息安全评估而言是非常重要的。

(2) 关联规则发现。使用关联规则发现方法可以从 Web 访问事务集中，找到一般性的关联知识。

(3) 序列模式发现。在时间戳有序的事务集中,序列模式的发现就是指找到那些如"一些项跟随另一个项"这样的内部事务模式。

(4) 分类。发现分类规则可以给出识别一个特殊群体的公共属性的描述。

(5) 聚类。可以从 Web Usage 数据中聚集出具有相似特性的那些客户。

### 3. Web 访问日志挖掘

Web 应用挖掘中,有一种是对 Web 访问日志(Web Log)进行分析和挖掘,Web 访问日志挖掘的基本流程包括如下步骤:

(1) 首先要对 Web Log 进行清洗、过滤和转换,从中抽取感兴趣的数据。

(2) 将资源的类型、资源的大小、请求的时间、在资源上停留的时间、请求者域名、用户、服务器状态作为多维数据立方体(Data Cube)的维变量,将对页面和文件请求次数等分别作为在这些维变量下的度量变量建立多维数据立方体(Data Cube)。

通过对数据立方体的切块、切片分析可以回答:哪些成分或特色被经常或偶尔使用,网络流量随时间的变化规律(按时、日、月等),用户在不同 Internet 域的分布情况,来自不同地区的用户在存取方式上是否有差异。

(3) 利用成熟的数据挖掘技术(如特征提取、分类、关联、预测、时间序列分析、趋势分析)进行 Web 流量分析、典型的事件序列和用户行为模式分析等,可以回答什么是典型的事件序列;用户的行为模式是什么;不同用户群在使用和行为上有什么差异;用户的行为是否随时间变化,怎么变化等问题。

通过分析 Web 访问日志能帮助理解用户的行为和 Web 结构,因此,可以改进 Web 页面的设计和 Web 应用程序,发现潜在的电子商务客户。

Web Log 分析还有助于建立针对个体的个性化 Web 服务。由于 Web Log 数据提供了用户访问 Web 页面的信息,因此 Web Log 信息可以与 Web 内容挖掘和 Web 结构挖掘集成起来,用于 Web 页面的等级划分、Web 文本的分类和多层次 Web 信息库的构造。

### 4. 应用挖掘的作用

通常在访问网站时,页面访问的顺序非常重要,以至于要把这个顺序作为一个整体来研究。当关联规则应用到这个序列时,就可以得到这次业务的顺序规则——比如先到主页,再到找工作列表,然后到联系方式。

这样的页面访问可以依照不同情况分为不同的类。这些类代表了不同的客户,比如老客户和新客户,浏览的客户和想购物的客户。不同的访问者访问同一个网站的目的是不一样的。比如,访问一个零售商网站的人的目的可能是购物或寻找就业机会。

Web 应用挖掘的好处主要有:

(1) 利用 Web 应用挖掘可以实现用户建模;

(2) 利用 Web 应用挖掘发现导航模式,从而改进 Web 站点的结构设计,实行个性化推销;

(3) 利用 Web 应用挖掘改进访问效率,改进服务器性能;

(4) 利用 Web 应用挖掘还可以进行个性化服务;

(5) 利用 Web 应用挖掘进行商业知识的发现;

(6) 利用 Web 应用挖掘进行用户移动模式的发现。

### 13.3.5 Web 日志分析与实例

#### 1. Web 日志数据概述

Web 访问日志数据具有如下基本特征:

(1) 动态变化性:随着时间的推移,访问信息连续不断产生并记录在日志文件中,因此数据集合具有较强的动态特征。

(2) 数据量大:由于访问记录实时更新,从时间跨度上趋于无限性,因此占用的数据总量是非常可观的,在综合网站中更是如此,每天的访问用户总量以及同时在线的用户量均是非常巨大的,产生的日志文件约有上百兆,访问次数达几十万甚至上百万。

(3) 多维性:Web 访问日志包含多个方面信息,如用户的 IP 地址、访问内容、访问方式、传输字节数、访问时间等,尽管不同格式日志文件记录的内容不尽相同,但是数据均具有多维性。

(4) 结构化程度较高:数据均按照确定的数据格式自动进行记录,并可按照一定规则进行相互转换,易于转换为关系数据库存储形式进行结构化处理。

(5) 包含大量琐碎数据:日志中大量因为网页下载而自动产生并记录后缀为 JPEG、GIF、SWF 等媒体文件信息以及 CSS 等样式文件,这些记录是与数据分析无关的。

由于用户访问网站而产生的 http 请求,这种动态环境中产生的信息即构成了连续不断的流式数据。这些数据信息存储在 Web 服务器访问日志中,其中记录了访问时间、访问客户端的 IP 地址、请求的 URI 地址、协议类型、传输字节数等内容。

Web 日志的记录格式包括如下常用字段:

① 每次访问者的客户端机器 IP 地址。

② 用户访问日期和时间,精确到秒。

③ 用户访问的网页名称。

④ 用户的本次访问请求是否成功的状态。

⑤ 传输文件的字节大小。

⑥ 引导用户访问到本站点的前驱 URL。

⑦ 访问者使用的浏览器版本和操作系统版本。

在 Web 日志分析中,把握动态变化趋势很重要,其内在规律、有用知识可能是稍纵即逝的,如用户访问趋势、访问热点变化、异常访问模式等。

热点访问的动态变化趋势,如连续查询几个版块半个小时内的访问总量,可以实时跟踪访问量最大的主题。同时这种热点可能随着时间的推移可能会转到另外一个主题,因此通过连续查询可发现主题的变化情况,可进一步分析产生变化的原因。

异常访问模式,是依据静态数据分析得到的先验知识,如用户的访问模式、访问频率范围等信息,连续查询某些版块或者地区的访问总量,可实时跟踪超出常规范围的聚集单元,因此可以及时发现异常访问,进而可以聚焦到该部分数据单元,进行下钻获取细粒度单元上

的异常情况。

流式数据多维查询与挖掘则是实现 Web 日志分析的有效工具。

**2. 面向 Web 日志分析的流式数据多维模型**

Web 日志分析主要关注时间维度、主题维度、用户维度等三个维度，在实际应用中可根据需求扩展其他分析维度，如访问状态、来源页面和用户代理等。

(1) 时间维度

时间维度是流式数据多维模型的基础，其中包含年、月、日、时、分、秒等多个时间粒度层次信息。传统的分析方法多关注数据的长期变化趋势，因此高层次时间粒度更为重要，如年、月、日等；而流式数据多维查询更多关注细粒度的数据动态变化情况，如时、分、秒等。

时间维度遵循多层次时间窗口模型约束，时间粒度层次与时间窗口长度的选择主要依据实际应用需求。如小时层次对应窗口长度为 24 小时，分钟层次对应窗口长度为 12 小时，秒钟层次对应数据窗口长度为 6 小时。

(2) 主题维度

综合网站可划分为滚动新闻、明星新闻、影视新闻等多个子版块，每个子版块关注不同的新闻内容，即不同主题。在具体实现过程中可依据用户访问的 URI 地址确定访问页面及关注主题，建立主题维度结构可划分为三个概念层次：①ALL 层次，代表所有主题的内容；②子版块层次，代表不同主题的内容；③页面层次，代表每个版块中不同的条目。

(3) 用户维度

用户维度属性取值代表不同地区的访问者，其中的维成员对应于 Web 访问日志中的访问主机，即访问客户端的 IP 地址。按照 IP 地址可确定访问用户所在的区域，如省、城市等。在建立用户概念层次过程中可采用适应性划分策略，仅选择重点关注的地区进行细粒度划分，而其他的区域则仅保留粗粒度事实即可，如重点关注"浙江地区"用户的访问情况，可在维度概念层次中建立城市级别的细粒度维成员，而其他地区则仅维持省级的维成员。建立概念层次结构可包含四个层次：ALL、国家、省、城市。

(4) 度量属性

度量属性和聚集函数均与分析目标息息相关，其中常见度量属性包括：

① 请求数：Web 访问日志的每一条记录对应一次请求，其中包括页面请求，也包括图片、Flash 等资源请求，所以一般打开一个页面会发送多个请求，根据网页设计的差异，请求数是页面浏览数的几倍。

② 页面浏览数：页面被打开(请求)的次数，是网站分析中最常见的度量。在下面的 Web 日志分析案例中，通过预处理将非页面访问的记录过滤，仅保留对页面文件的请求记录。对页面浏览数的聚集计算可用于发现用户的访问热点，访问异常等。

③ 传输数据量：传输数据量可用于统计网站的流量，以及衡量不同用户以及不同周期内的访问情况，需要将所有请求的传输字节数相加得到结果。

在分析过程中还可以基于度量值进行求和、比例、平均等聚集计算获得不同维度视角或者数据粒度层次上的信息，进而为深层次综合分析与挖掘提供更丰富的信息支持。

### 3. Web 日志数据的多维查询

在 Web 日志分析中，管理者或分析者经常需要获取某一特定维度视角或数据粒度层次上的聚集信息，这种即席方式的多维查询通常由用户依据需求即时定义，一次性执行并返回查询结果。为了支持快速查询计算，需在主存中维持预先计算的流立方体，用于存储兴趣视图中的数据单元。下面以流式数据三维视图(小时，城市，视频版块)为例，进行 Web 日志多维查询。

切片(切块)与钻取查询是应用广泛的多维分析操作，用于获取选择不同粒度层次上查询范围内满足条件的数据单元，以便发现高峰访问时段、重点关注的视频版块和主要用户所在城市或地区的分布。

侯东风博士生对 Web 日志进行了查询，举例如下：

(1) Q1：对上午 6 时至 12 时，查询每个小时的页面浏览数，查询视图为(小时，*，*)，切块查询条件定义在时间维度上。

Q1 查询结果如表 13.2 所示，结果表明，网站的高峰访问时间段为上午“9～10 时”，而其他时段则相对较少。

**表 13.2　Q1 查询结果**

| 时　间 | 页面浏览数 | 时　间 | 页面浏览数 |
|---|---|---|---|
| 5～7 时 | 953 | 9～10 时 | 5122 |
| 7～8 时 | 1080 | 10～11 时 | 4270 |
| 8～9 时 | 3295 | 11～12 时 | 1945 |

(2) Q2：对工作时间 7 时至 12 时之间，查询主要的子版块的页面浏览数，即在上面查询的基础上，下钻到子版块的查询。查询视图为(小时，*，子版块)，在时间维度和主题维度上定义切块查询。

Q2 查询结果如表 13.3 所示，表明上午工作时间的访问多集中在“滚动新闻”，而其他版块相对较少。

**表 13.3　Q2 查询结果**

| 时　间 | 子　版　块 | 页面浏览数 |
|---|---|---|
| 7～12 时 | 滚动新闻 | 1945 |
| 7～12 时 | 图片新闻 | 537 |
| 7～12 时 | 综艺新闻 | 899 |
| 7～12 时 | 明星新闻 | 374 |
| 7～12 时 | 音乐新闻 | 342 |
| 7～12 时 | 人物访谈 | 308 |

(3) Q3：对晚间 18 时至 22 时之间，查询主要省份的页面浏览数，如北京，上海，广东，

湖南，江苏，浙江，辽宁，重庆。查询视图为(小时，省，*)，在时间维度和用户维度上定义切块查询。

Q3 查询目的在于发现访问者的区域分布情况，查询结果如表 13.4 所示，表明热点访问省级区域为“浙江”和“北京”。

**表 13.4　Q3 查询结果**

| 时　　间 | 省　　份 | 页面浏览数 |
|---|---|---|
| 8～22 时 | 北京 | 5454 |
| 8～22 时 | 上海 | 518 |
| 8～22 时 | 广东 | 1415 |
| 8～22 时 | 湖南 | 334 |
| 8～22 时 | 江苏 | 705 |
| 8～22 时 | 浙江 | 10101 |
| 8～22 时 | 辽宁 | 261 |
| 8～22 时 | 重庆 | 165 |

若需进一步探查浙江省主要城市的访问者区域分布情况，则需从用户维度下钻到城市层次进行计算，即执行 Q4 查询。

(4) Q4：对浙江省主要城市，查询 18 点至 22 点之间的页面浏览数，如杭州市，宁波市，温州市，绍兴市，嘉兴市，金华市。查询视图为(小时，城市，*)，在时间维度和用户维度上定义切块查询。

查询结果如表 13.5 所示，结果表明网站的访问者大部分来自于杭州市和嘉兴市，而其他城市较少。

**表 13.5　Q4 查询结果**

| 时　　间 | 省　　份 | 城　　市 | 页面浏览数 |
|---|---|---|---|
| 18～22 时 | 浙江 | 杭州市 | 3381 |
| 18～22 时 | 浙江 | 宁波市 | 159 |
| 18～22 时 | 浙江 | 温州市 | 120 |
| 18～22 时 | 浙江 | 绍兴市 | 87 |
| 18～22 时 | 浙江 | 嘉兴市 | 1134 |
| 18～22 时 | 浙江 | 金华市 | 33 |

从以上 Web 日志数据的多维查询分析可以得出的结论是：①网站上的页面浏览的高峰访问时段是上午 9～10 时。②工作时间访问最多的是滚动新闻。③晚间访问网站最多的省份是浙江和北京，其中浙江省内访问最多的用户主要在杭州市和嘉兴市。

#### 4. 连续查询热点访问信息或异常现象

在瞬息万变的形势下，数据的动态变化趋势对决策支持具有现实意义，如发现热点访问信息或异常现象等。

分析访问热点或异常情况，可分别定义连续查询要求如下：

（1）Q5：连续跟踪最近 1 小时内的每个版块的页面浏览总数变化趋势。查询视图为（分钟，*，子版块），时间窗口为 60 分钟，查询条件分别对应不同版块，包括明星新闻、滚动新闻、图片新闻、影视新闻。

Q5 连续的查询结果表明，“滚动新闻”版块的访问量较高，在 10～24 时间段，访问量相对稳定在 900 次左右。比其他子版块要高出 400～600 次，并且在上午 9 时监测到一个高峰访问周期，达到 1600 次，比平均值高出近一倍，这是一个异常情况。

（2）Q6：连续跟踪最近 1 小对内不同城市的访问量变化情况，包括浙江省的杭州市，嘉兴市，丽水市。查询视图为（分钟，城市，*），时间窗口为 60 分钟，查询条件定义在用户维度上。三城市不同时间访问页面次数对比表如表 13.6 所示。

表 13.6　浙江省三城市不同时间访问页面次数对比表

| 时　间 | 杭　州　市 | 嘉　兴　市 | 丽　水　市 |
|---|---|---|---|
| 02 时 | 50 | 150 | 0 |
| 04 时 | 30 | 50 | 0 |
| 06 时 | 80 | 50 | 0 |
| 08 时 | 100 | 180 | 0 |
| 10 时 | 210 | 220 | 1100 |
| 12 时 | 100 | 190 | 700 |
| 14 时 | 150 | 50 | 800 |
| 16 时 | 650 | 80 | 700 |
| 18 时 | 100 | 150 | 850 |
| 20 时 | 180 | 160 | 920 |
| 22 时 | 230 | 30 | 600 |
| 24 时 | 100 | 500 | 800 |

Q6 连续的查询是针对杭州市、嘉兴市和丽水市，进行对比分析结果表明：

丽水市在 8～24 时，平均访问量有 800 次，大大高于其他两个城市（相当于 4～6 倍）。杭州市平均访问量 200 次，嘉兴市平均访问量 120 次。

丽水市在上午 10 时，页面浏览数的访问量达到最高峰 1100 次。杭州市在下午 16 时达到最高峰，访问量是 700 次。嘉兴市在上午 9 时达到最高峰，访问量是 400 次。但丽水市清晨时的访问量是 0。可见丽水市在浙江省是一个特例。

（3）Q7：连续跟踪主要省份对特定版块的访问量变化情况，查询视图为（分钟，省，滚动

新闻)，时间窗口为 60 分钟，查询条件定义在用户维度上，用户分别对应浙江省、江苏省和上海市，主题维度的子版块定为“滚动新闻”。

Q7 连续的查询结果，表明浙江省对“滚动新闻”版块关注较多，在 0～7 小时内，平均访问量有 30 次左右；在 8～24 小时内，平均访问量有 70 次左右。而江苏省和上海市关注较少，这两地的访问量差不多，在 0～7 小时内，平均访问量只有 5 次左右；在 8～24 小时内，平均访问量只有 15 次左右。

(4) Q8：连续跟踪浙江省对主题维度的不同版块访问的变化情况，包括滚动新闻，影视新闻，近期热点。查询视图为(分钟，浙江省，子版块)，时间窗口为 60 分钟，查询条件定义在主题维度的子版块上。

Q8 的查询结果显示了一个有趣的现象：

“近期热点”版块的页面浏览数在工作时间关注较多，在 8～18 小时内，平均访问量有 90 次左右。

“滚动新闻”版块在这时段，平均访问量只有 60 次左右。但是，“滚动新闻”版块在休息时间(18～23 时)关注较多，平均访问量有 80 次左右，在第 24 时达到了高峰，访问量达到 180 次。

(5) Q9：最近 10 分钟，查询杭州市对几个主要版块访问的变化趋势，检查是否有异常现象。对滚动新闻，明星新闻，图片新闻的查询视图为(秒钟，杭州市，子版块)，时间窗口长度为 600 秒钟。查询条件定义在主题维度上。

Q9 的查询结果表明，在短期内对几个版块的访问分布都较均匀，其中“滚动新闻”在 10～18 时，访问量在 10 次左右，仅出现较少的峰值，但在 9 点钟左右，10 分钟内“滚动新闻”的页面浏览数的访问量达到了 52 次，是平均访问量的 5 倍，这是一个异常现象。

**小结：**

从以上多种连续查询中，可以概括为：热点访问的时间段、新闻版块或用户地区，主要体现在页面浏览数的平均值比其他的高。而异常现象表现在一段时间内的访问量是平均访问量的数倍之多。

通过“热点访问”和“异常现象”的分析，为网站的页面的进一步开发，提供了有效的决策支持。

# 习 题 13

1. 文本挖掘的概念是什么？
2. 文本挖掘与数据挖掘有什么不同？
3. 文本挖掘的主要任务是什么？
4. 文本特征包含什么内容？
5. 如何形式化表示文本特征？
6. 文本特征提取的基本算法过程是什么？
7. 说明文本挖掘的功能层次内容。

8. 说明文本关联分析的基本思想。
9. 说明文本的层次聚类法和平面划分法的基本思想与区别。
10. Web信息有什么特点？
11. Web挖掘与文本挖掘有什么区别和联系？
12. 简述Web数据挖掘的意义。
13. 说明Web挖掘的分类及含义。
14. Web内容挖掘的目的是什么？
15. “召回”与“精度”的含义是什么？它们之间的关系是什么？
16. 简述Web结构挖掘的主要任务和目的。
17. 什么是中枢站点和权威站点？
18. Web应用挖掘的意义是什么？
19. Web应用挖掘中的常用技术有哪些？
20. 为什么对Web日志的分析采用流式数据的多维流立方体模型？
21. 通过实例说明Web日志数据的多维切片与钻取分析，能发现什么？
22. 在连续查询中如何发现“热点访问”和“异常现象”。

# 参 考 文 献


1. W H Inmon. 数据仓库. 北京：机械工业出版社，2000.
2. W H Inmon，K Rudim. 数据仓库管理. 北京：电子工业出版社，2000.
3. P Ponniah. 数据仓库基础. 北京：电子工业出版社，2004.
4. J Bischoff，T Alexander. 数据仓库技术. 北京：电子工业出版社，1998.
5. E Thomsen. OLAP 解决方案. 2 版. 北京：电子工业出版社，2004.
6. Spofford G，Harinath S. MDX 解决方案. 2 版. 北京：清华大学出版社，2008.
7. G S Linoff，S Micheal，J A Berry. Web 数据挖掘. 北京：电子工业出版社，2004.
8. J Han，M Kamber. 数据挖掘概念与技术. 北京：机械工业出版社，2001.
9. M Kantardzic. 数据挖掘——概念、模型、方法和算法. 北京：清华大学出版社，2003.
10. M H Dunham. 数据挖掘教程. 北京：清华大学出版社，2005.
11. R S Mchalski，J G Garbonell，T M Mitchell. 机器学习实现人工智能的途径. 北京：科学出版社，1992.
12. 陈文伟，黄金才. 数据仓库与数据挖掘. 北京：人民邮电出版社，2004.
13. 陈文伟，黄金才，赵新昱. 数据挖掘技术. 北京：北京工业大学出版社，2002.
14. 陈文伟，廖建文. 决策支持系统及其开发. 3 版. 北京：清华大学出版社，2008.
15. 陈文伟. 决策支持系统教程. 2 版. 北京：清华大学出版社，2010.
16. 陈文伟，陈晟. 知识工程与知识管理. 北京：清华大学出版社，2010.
17. 陈文伟. 智能决策技术. 北京：电子工业出版社，1998.
18. 陈文伟. 数据仓库与数据挖掘教程. 北京：清华大学出版社，2006.
19. 徐洁磐. 数据仓库与决策支持系统. 北京：科学出版社，2005.
20. 元昌安. 数据挖掘原理与 SPSS Clementine 应用宝典. 北京：电子工业出版社，2009.
21. 廖芹，郝志峰，陈志宏. 数据挖掘与数学建模. 北京：国防工业出版社，2010.
22. 姚家奕，等. 多维数据分析原理与应用实验教程. 北京：电子工业出版社，2007.
23. 张云涛，龚玲. 数据挖掘原理与技术. 北京：电子工业出版社，2004.
24. 王珊，等. 数据仓库技术与联机分析处理. 北京：科学出版社，1998.
25. 徐立本. 机器学习引论. 长春：吉林大学出版社，1991.
26. 洪家荣. 归纳学习——算法理论应用. 北京：科学出版社，2001.
27. 史忠植. 知识发现. 北京：清华大学出版社，2002.
28. 王正志，等. 进化计算. 长沙：国防科技大学出版社，2000.
29. 陈国良，等. 遗传算法及其应用. 北京：人民邮电出版社，1999.
30. 刘清. Rough 集及 Rough 推理. 北京：科学出版社，2001.
31. 王国胤. Rough 集推理论与知识获取. 西安：西安交通大学出版社，2001.
32. 陈文伟，黄金才，等. 决策支持系统新结构体系. 管理科学学报，1998(9).
33. 钟鸣，陈文伟. 示例学习的抽象信道模型及其应用. 计算机研究与发展，1992，29(1).
34. 钟鸣，陈文伟. 示例学习算法 IBLE 和 ID3 的比较研究. 计算机研究与发展，1993，30(1).
35. 陈文伟，等. 数据开采技术研究. 清华大学学学报(自然科学版)，1998，38(2).
36. 马建军，陈文伟. 关于集合理论的 KDD 方法. 计算机应用研究，1997，14(3).
37. 陈文伟，黄金才. 基于神经网络的模糊推理. 模糊系统与数学，1996(4).
38. 陈文伟，挖掘变化知识的可拓数据挖掘研究. 中国工程科学，2006，8(11)：70-73.
39. 陈文伟，杨春燕，黄金才. 可拓知识与可拓知识推理. 哈尔滨工业大学学报，2006，38(7)：1094-1096.

40. 陈文伟. 基于本体的可拓知识链获取. 智能系统学报，2007,2(6)：68-71.
41. 陈文伟,黄金才. 从数据挖掘到可拓数据挖掘. 智能技术，2006,1(2)：50-52.
42. 徐立本,陈文伟主编. 机器学习论文集. 国防科技大学学报(增刊)，1995(17).
43. 赛英,陈文伟,等. 从数据库中发现知识的方法研究与应用. 管理科学学报，1999 年 9 月.
44. 陈文伟,张帅. 经验公式发现系统 FDD. 小型微型计算机系统，1999 年 6 月.
45. 马建军,陈文伟. 关于集合理论的 KDD 方法. 计算机应用研究，1997,14(3).
46. 赵新昱,陈文伟,何义. 基于算子空间的公式发现算法研究. 国防科技大学学报. 2000,22(4).
47. 赵新昱,陈文伟. 基于遗传建模的公式发现研究. 计算机工程与科学，2000,22(5).
48. 邹雯,陈文伟. 一种新的遗传分类器学习系统(GCLS). 96' 人工智能进展,1996.
49. 陈文伟,钟鸣,赵东升. 示例学习的信息理论以及逻辑公式的生成. 中国机器学习 93' 论文集,1993.
50. 陈文伟. 可拓学与智能科学、信息科学. 香山科学会议(第 271 次会议),2005. 12.
51. 陈文伟,黄金才. 数学进化中矛盾问题的解决方法. 智能技术学报,2010. 1.
52. 陈文伟主编. 数据开采与数据仓库论文集. 信息与决策系统，1998,3(1).
53. 陈文伟,等. 数据仓库与决策支持系统. 计算机世界,1998. 6. 15.
54. 高人伯,陈文伟. 数据仓库和 OLAP 的数据组织. 计算机世界,1998. 6. 15.
55. 黄金才,陈文伟,陈元. 数据仓库中的元数据. 计算机世界,1998. 6. 15.
56. 陈元,陈文伟. OLAP 的多维数据分析. 计算机世界,1998. 6. 15.
57. 陈文伟,等. 综合决策支持系统. 计算机世界,1998. 6. 15.
58. 陈文伟,等. 数据开采与知识发现综述. 计算机世界,1997. 6. 30.
59. 陈文伟,钟鸣. 数据开采的决策树方法. 计算机世界,1997. 6. 30.
60. 马建军,陈文伟. 数据开采的集合论方法. 计算机世界,1997. 6. 30.
61. 邹雯,陈文伟. 数据开采中的遗传算法. 计算机世界,1997. 6. 30.
62. 王珊,罗立. 从数据库到数据仓库. 计算机世界,1996. 7. 15.
63. 王珊. 数据仓库、联机分析处理、数据挖掘——基于数据库技术的 DSS 解决方案. 计算机世界,1997. 1. 6.
64. 陈文伟，王朝霞. 神经元网络专家系统中学习算法. 中国机器学习 91' 论文集,1991.
65. 陈文伟,等. 示例学习的信息理论以及逻辑公式的生成. CMLW'93,中国机器学习 93. 电子工业出版社，1993.
66. 陈文伟,等. 可视化机器发现的研究. 国防科技大学学报,(增刊),1995(17).
67. 陈文伟，杨桂聪. NEEST 神经元网络专家系统工具. 神经网络 1991 年学术大会论文集，1991.
68. 陈文伟. 决策支持系统语言的设计和开发. 全国程序设计语言发展与教学会议论文集，1993.
69. 陈文伟,黄金才,毕季明. 解决矛盾问题的可拓模型与可拓知识的研究. 数学的实践与认识,2009,39(4).
70. 陈文伟. 论新常数 $\mu$、$\theta$ 和新公式 $\pi=\frac{1}{2}e^{\theta}$,高等数学研究,2009,12(6).
71. 陈文伟. 数据挖掘的可拓知识与元知识,中国人工智能进展,2007：942-946.
72. 陈文伟,等. 数据仓库的可拓决策分析工具,中国人工智能进展,2001：1085-1088.
73. 陈文伟,黄金才. 属性约简与数据挖掘的可拓变换与可拓知识的表示,重庆工学院学报,2007,21(7)：1-4.
74. 陈文伟,黄金才,陈晟. 数学进化中知识发现方法的研究,智能系统学报,2001,6(4).
75. 陈文伟,黄金才,毕秀明. 适应变化环境的元知识的研究,智能系统学报,2009,4(4).
76. 李凡,等. 关于文本特征抽取新方法的研究. 清华大学学报(自然科学版)，2001,41(7).
77. 侯东风,等. 数据立方体计算方法研究综述. 计算机科学，2008,35(10).

78. 王颖楠,等. Web挖掘技术. 吉林工学院学报，2002,23(1).
79. 韩家炜,等. Web挖掘研究. 计算机研究与发展，2001,38(4).
80. 陈莉,焦李成. Web数据挖掘研究现状及最新进展. 西安电子科技大学学报(自然科学版)，2001,28(1).
81. 胥桂仙,等. 文本挖掘中的特征表示及聚类方法. 吉林工学院学报，2002,23(3).
82. 黄金才. 网络环境下决策资源共享与决策支持系统快速开发环境研究. 国防科技大学博士学位论文，2001. 10.
83. 赵新昱. 模型规范化与多主体域组织模型研究. 国防科技大学博士学位论文，2001. 7.
84. 陈元. 基于分类模型的知识发现过程研究. 国防科技大学博士学位论文，2002. 3.
85. 赛英. 粗糙集扩展模型及其在数据挖掘中的应用研究. 国防科技大学博士学位论文，2002. 6.
86. 徐振宁. 基于本体的 Web 数据语义信息的表示与处理方法研究. 国防科技大学博士学位论文，2002. 9.
87. 戴超凡. 数据仓库中数据志跟踪的理论与方法研究. 国防科技大学博士学位论文，2002. 12.
88. 王长缨. 多 Agent 协作团队的学习方法研究. 国防科技大学博士学位论文，2004. 6.
89. 侯东风. 流式数据多维建模与查询关键技术研究. 国防科技大学博士论文,2010. 12.
90. R C Barquin，H A Edelstein. Planning and Designing the Data Warehouse. Prentice Hall PTR，1997.
91. R S Michalski，et al. Machine Learning an Artificial Intelligence Approach. Morgan Kaufmenn,1986.
92. J R Quinlan. Induction of Decision Trees，Machine Learning,1986;1(1).
93. J R Quinlan. $C_{4.5}$：Program for Machine Learning. Morgan Kaufmenn,1993.
94. J Cendrowska. PRISM：An Algorithm for inducing modular rules. Int. J. Man-Machine studies，1987，27.
95. P Adriaans，D Zantinge. Data Mining，Addison-Wesley，1996.
96. U Fayyad，G Pietetsky-Shapiro，P Smyth. From Data Mining to Knowledge Discovery in Database，AAAI，0738-4602，1996.
97. R Yasdi. Learning Classification Rules from Database in the Context of Knowledge Acquisition and Representation，IEEE TKDE，1991,3(3).
98. U Fayyad，R Uthurusamy. Proc. of the First Int. Conf. on Knowledge Discovery and Data Mining. AAAI Press，1995.
99. Zou Wen，Chen Wenwei. A New Genetic Classifier Learning System (GCLS). Genetic Programming Conference(GP'97),1997.
100. Shi Zhongzhi. Principles of Machine Learning,International Academic Publishers,1992.
101. U Fayyad，G Piatetsky-Shapiro. Advances in Knowledge Discovery and Data Mining，AAAI Press,1996.
102. U Fayyad,R Uthurusamy. Data Mining and Knowledge Discovery in Database,Communications of the ACM,1996,39(11).